DUDLEY KNOX LIBRARY
NAVAL
MONTEREY

NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

A MATCHED FILTER ALGORITHM FOR
ACOUSTIC SIGNAL DETECTION

by

Dorsett Weston Jordan

June 1985

Thesis Advisor: R. Panholzer

Approved for public release; distribution is unlimited

T222834

SECURITY CLASSIFICATION OF THIS PAGE *(When Data Entered)*

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE *(and Subtitle)* A Matched Filter Algorithm for Acoustic Signal Detection		5. TYPE OF REPORT & PERIOD COVERED Master's Thesis; June 1985 6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR*(s)* Dqrsett Weston Jordan		8. CONTRACT OR GRANT NUMBER*(s)*
9. PERFORMING ORGANIZATION NAME AND ADDRESS Naval Postgraduate School Monterey, California 93943-5100		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Naval Postgraduate School Monterey, California 93943-5100		12. REPORT DATE June 1985 13. NUMBER OF PAGES 221
14. MONITORING AGENCY NAME & ADDRESS*(if different from Controlling Office)*		15. SECURITY CLASS. *(of this report)* UNCLASSIFIED 15a. DECLASSIFICATION/ DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT *(of this Report)*

Approved for public release; distribution is unlimited

17. DISTRIBUTION STATEMENT *(of the abstract entered in Block 20, if different from Report)*

18. SUPPLEMENTARY NOTES

19. KEY WORDS *(Continue on reverse side if necessary and identify by block number)*

Analog Filter; Digital Filter; Bilinear Transform; Space Shuttle; Auxiliary Power Unit; INTEL 2920 Analog Signal Processor; Adaptive LInear Combiner; Adpative Transversal Filter

20. ABSTRACT *(Continue on reverse side if necessary and identify by block number)*

This thesis is a presentation of several alternative acoustic filter designs which allow Space Shuttle payload experiment initiation prior to launch. This initiation is accomplished independently of any spacecraft services by means of a matched band-pass filter tuned to the acoustic signal characteristic of the Auxiliary Power Unit (APU) which is brought up to operating RPM's approximately five minutes prior to launch.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

S/N 0102-LF-014-6601

SECURITY CLASSIFICATION OF THIS PAGE *(When Data Entered)*

These alternative designs include an analog filter built around operational amplifiers, a digital IIR design implemented with an INTEL 2920 Signal Processor, and an Adaptive FIR Weiner design. Working prototypes of the first two filters are developed and a discussion of the advantage of the 2920 digital design is presented.

Approved for public release; distribution is unlimited.

A Matched Filter Algorithm for Acoustic Signal Detection

by

Dorsett Weston Jordan
Lieutenant, United States Navy
B.S., University of Colorado, 1977

Submitted in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL
June, 1985

Thesis
J754
c.1

ABSTRACT

This thesis is a presentation of several alternative acoustic filter designs which allow Space Shuttle payload experiment initiation prior to launch. This initiation is accomplished independently of any spacecraft services by means of a matched band-pass filter tuned to the acoustic signal characteristic of the Auxiliary Power Unit (APU) which is brought up to operating RPM's approximately five minutes prior to launch.

These alternative designs include an analog filter built around operational amplifiers, a digital IIR design implemented with an INTEL 2920 Signal Processor, and an Adaptive FIR Weiner design. Working prototypes of the first two filters are developed and a discussion of the advantage of the 2920 digital design is presented.

TABLE OF CONTENTS

I. INTRODUCTION---------- 9

A. BACKGROUND---------- 9

B. STATEMENT OF GOAL---------- 10

C. DISCUSSION OF THE GENERAL SOLUTION ALGORITHM--- 12

II. THE AUXILIARY POWER UNIT (APU) AND THE SHUTTLE PAYLOAD BAY PRE-LAUNCH ACOUSTIC ENVIRONMENT------- 15

A. APU DESCRIPTION---------- 15

B. APU MISSION DUTY CYCLE---------- 15

C. STS PAYLOAD BAY ACOUSTIC ENVIRONMENT MEASUREMENT INSTRUMENTATION---------- 19

D. STS PAYLOAD BAY PRE-LAUNCH ACOUSTIC ENVIRONMENT---------- 21

III. ANALOG FILTER DESIGN THEORY AND IMPLEMENTATION---- 26

A. PRACTICAL DEVELOPMENT OF AN ANALOG BAND-PASS FILTER---------- 26

1. Low-Pass to Band-Pass Frequency Transformation---------- 28

2. 600 Hz Elliptic Band-Pass Filter Design---- 32

3. Analog Band-Pass Filter Simulation--------- 37

B. HARDWARE IMPLEMENTATION---------- 41

1. Biquad Analog Band-Pass Filter---------- 41

2. Follow-Up Logic Circuitry---------- 44

C. ANALOG BAND-PASS FILTER IMPLEMENTATION RESULTS- 47

IV. DIGITAL FILTER DESIGN---------- 51

A. ANALOG BAND-PASS TO DIGITAL BAND-PASS FILTER DESIGN---------- 54

B. DIGITAL BAND-PASS FILTER SIMULATION---------- 60

C. DIFFERENCE EQUATION REPRESENTATION---------- 63

D. DIFFERENCE EQUATION IMPLEMENTATION SIMULATION-- 66

1. Digital Filter Impulse Response---------- 66

2. Digital Filter Frequency Response---------- 69

E. ANTI-ALIASING FILTER---------- 70

V. THE INTEL 2920 ANALOG SIGNAL PROCESSOR---------- 72

A. OVERVIEW---------- 72

B. 2920 FUNCTIONAL DESCRIPTION---------- 75

C. THE SDK-2920 DEVELOPMENT SYSTEM---------- 76

D. A 2920 DIGITAL FILTER IMPLEMENTATION---------- 80

1. 2920 Assembly Language Program---------- 80

2. 2920 Hardware Implementation---------- 82

E. 2920 DIGITAL FILTER IMPLEMENTATION RESULTS----- 84

VI. ALTERNATIVE FILTER CONCEPTS---------- 88

A. A WIENER FILTER DESIGN--THE ADAPTIVE LINEAR COMBINER---------- 89

1. Theoretical Foundations---------- 90

2. A Software Simulation---------- 95

B. AN ANALOG SPEECH PROCESSING SCHEME---------- 98

VII. CONCLUSION---------- 101

APPENDIX A: APU HOTFIRE TEST RESULTS (QUICK LOOK DATA)- 108

APPENDIX B: POWER SPECTRAL DENSITY PLOTS OF THE SHUTTLE CARGO BAY PRE-LAUNCH ACOUSTIC ENVIRONMENT-- 121

APPENDIX C: FORTRAN PROGRAM ABPDBP AND PROGRAM OUTPUT-- 146

APPENDIX D: FORTRAN PROGRAM ABPFR---------- 161

APPENDIX E: FORTRAN PROGRAM DBPFR---------- 164

APPENDIX F: FORTRAN PROGRAMS S22I (WITH OUTPUT) AND S22IG---------- 167

APPENDIX G: FORTRAN PROGRAMS S22F (WITH OUTPUT) AND S22FG (WITH GRAPHICAL OUTPUT)---------- 176

APPENDIX H: 2920 ASSEMBLY LANGUAGE PROGRAM SM2X2------- 188

APPENDIX I: FORTRAN PROGRAM CTRANS2 AND OUTPUT--------- 194

APPENDIX J: FORTRAN PROGRAM FIR4---------- 200

APPENDIX K: FORTRAN PROGRAM FIR4SIM---------- 216

LIST OF REFERENCES---------- 218

INITIAL DISTRIBUTION LIST---------- 220

ACKNOWLEDGMENT

Many persons aided me in the development of this thesis and I would be remiss in not acknowledging the help and support of those whose contributions were invaluable.

Vicente Garcia offered early insight and suggested avenues to a solution. Bob Yamamuro of Aerospace Corporation assisted me in the initial data collection and unselfishly spent several evenings of his own time in support of my efforts. Austin Boyd gave assistance with crucial data analysis during a difficult time in his life. Professor Rudy Panholzer was a talented teacher and supportive advisor throughout my stay at NPS. Dr. Bharat Madan was indispensable to me as an expert in digital filters and all aspects of the 2920 design implementation. He gave me many hours of wise counsel. David Rigmaiden was and is a superb and supremely patient lab technician. Thanks to all the NPS Get Away Special team members whose motivation and spirit kept me honest. And finally thanks and love to Linda Duncan-Hille who buoyed my spirits at all the right moments with her abundant joie de vivre.

I am especially grateful for the help that each and every one of you gave me during this research.

I. INTRODUCTION

A. BACKGROUND

In April of 1981 a new era of space exploration was opened for mankind when the Space Shuttle successfully returned to Earth after its first orbital mission. Notwithstanding the significant capabilities which this new and radically different concept in space transportation has afforded traditionally large and well funded government and industrial agencies, the Space Shuttle has also ushered in the age of the small experimenter with limited access to funds who has an idea which needs to be tested in space and who now has a means of seeing that test realized.

The Get Away Special (GAS) Program was developed by the National Aeronautics and Space Administration (NASA) to afford qualifying teams of experimenters the opportunity to launch and orbit GAS payloads on a not-to-interfere space available basis during regularly scheduled Shuttle missions.

The main purpose for Shuttle missions is the conveyance into space and deployment of large, sophisticated instruments into Earth orbit. Because many primary mission payloads do not occupy the entirety of the Shuttle payload bay there is frequently opportunity for the additional

deployment of GAS payloads during Shuttle missions [Ref. 1: pp. 8-9].

One significant requirement demanded of GAS experimenters is that each payload be self-contained within its standard size (5.0 cubic foot) NASA provided canister. Thus it is expected that each GAS experiment include provisions within its own confines for all electrical power, heating elements, data control and storage, and so on. In short, GAS payloads may not draw upon any Shuttle services for their normal operation beyond an on-off switch control which may be thrown locally by an astronaut should the Shuttle timeline of events allow for such intervention.

This requirement for a completely self-contained experiment necessitates a well-planned execution of tasks performed under automated control. Although the concession for astronaut involvement in experiment initiation is provided, during some critical periods it is not feasible for astronauts to tend to GAS payload requirements, e. g., during launch and landing phases. At these times it then becomes necessary to design for complete automation of GAS experimental control, to provide most significantly for independent initiation of experimentation.

B. STATEMENT OF GOAL

It is the purpose of this thesis to consider a design algorithm to accomplish an automated initiation of GAS

experimentation based solely upon the passive detection of a well-defined event in the evolution of the Shuttle timeline of operations. Specifically it is desired that an experiment be undertaken to measure the acoustical and vibrational environment present in the Shuttle payload bay from prior to Space Shuttle Main Engine (SSME) ignition (at approximately T-6 seconds) until the Orbiter has exited the Earth's atmosphere (at approximately T+2 minutes).

Previous GAS missions have been deployed to accomplish this same mission goal. In these previous missions the solution to the problem of independent experiment initiation was addressed by way of a simple sound pressure level (SPL) sensor used to detect the unmistakable roar of SSME's as they ignite at T-6 seconds. By this method the sizable impulse thus generated can easily be used to signal experiment initiation. In this way power is applied to data collection and storage elements whose purpose is to record the information relevant to the mission of the experiment.

The obvious shortcoming in this experimental scheme is that it uses as its prime mover the very noise and vibration which it presumes to measure. Though the delay caused by impulse generation and transmission via the SPL sensor may be minimal, it nonetheless requires a finite amount of time for tape transport transients to settle and begin the recording of meaningful data. This delay has been reported to be as much as several seconds and so can be seen to be a

significant gap in any serious analysis of the acoustical and vibrational transients which must accompany ignition and launch.

One means of lessening the impact of transient delay is to substitute a solid state data recorder for the traditional magnetic tape variety of instrumentation data recorders. This idea is being investigated by another team of researchers at the Naval Postgraduate School for incorporation into a deployable GAS mission canister. Despite the advantage which a solid state recorder will afford toward minimizing the transient delay prior to meaningful data collection, it can never completely eliminate the transient effect which accompanies any scheme which is trying to measure the same signal which it also uses for experiment initiation. There is a causal imperative here which is inescapable.

It is the purpose of this thesis then to develop a means of experiment initiation which will allow data collection to commence well before SSME ignition. In this manner we will be allowed a full measure of the acoustical and vibrational environment which is present in the Shuttle payload bay during launch.

C. DISCUSSION OF THE GENERAL SOLUTION ALGORITHM

The timeline of Shuttle events prior to launch includes the turn-on of Auxiliary Power Units (APU) at approximately

T-5 minutes. These units are essentially jet engines which provide for hydraulic power of Shuttle airfoil control surfaces during the atmospheric phases of launch and landing. They are designed to operate at very high (72,000) RPM's, but also generate a very specific acoustic signature in the audio range of the spectrum during normal operation. If it is possible to detect this APU acoustic signature during the pre-launch sequence of events and to discriminate this signature from among the various other acoustic events which may also occur during the pre-launch phase, then it may be possible to signal experiment initiation by detection of this event.

The emphasis of this thesis will be to describe the nature of the APU acoustic signature and to develop a matched filter which is tuned to its characteristics. It should be emphasized from the beginning that this it is not my intention to develop a classical "matched filter" which rigorously conforms to that definition. I do not have the uncontaminated APU acoustic signature data at my disposal which would allow that sort of an analysis. Rather it is my intention to examine the APU signature in the Shuttle cargo bay environment and to develop a filter which is "matched" to that contaminated signature. It is expected that an extremely narrow (high Q) band-pass filter will accomplish this goal.

Several different schemes for accomplishing a matched band-pass filter design will be discussed. Nominally this will include an analog design built around operational amplifiers and a digital design of Infinite Impulse Response (IIR) implemented with an INTEL 2920 Signal Processor. This latter configuration will be derived from a cascaded IIR design which results from the Bilinear transformation of the analog filter. The difference equation representation of the digital filter transfer function will form the basis for the 2920 design. In addition I will discuss further design alternatives including an Adaptive Finite Impulse Response (FIR) Weiner filter and an idea for a design centered about a speech processing algorithm. Advantages and disadvantages of each approach will be discussed.

Ultimately one design will be chosen for integration within the GAS experiment just described and scheduled for launch in an upcoming Shuttle mission.

II. THE AUXILIARY POWER UNIT (APU) AND THE SHUTTLE PAYLOAD BAY PRE-LAUNCH ACOUSTIC ENVIRONMENT

A. APU DESCRIPTION

The Auxiliary Power Unit (APU) was developed by the Sundstrand Corporation under contract from Rockwell International, the prime contractor for the Space Shuttle. Each Shuttle Orbiter is equipped with three complete APU's and associated hydraulic systems. Each APU and its hydrazine fuel system is independent of the other two during normal operations. However, there are cross-ties between hydraulic systems which allow any two APU's to pick up the load from a third should it fail during operation.

B. APU MISSION DUTY CYCLE

Figure 2.1 is a representative diagram of a typical APU Mission Duty Cycle for an entire Shuttle mission [Ref. 2]. It is expected that a minimum of two restarts from a cold condition will be typical in a mission. The baseline duty cycle calls for 81.1 minutes of APU operation at various power levels from 8.0 horsepower to its maximum rated 135.0 horsepower. This includes launch, de-orbit, re-entry and landing phases and so includes operation at all altitudes corresponding to extremes of airfoil atmospheric resistance and Orbiter speed.

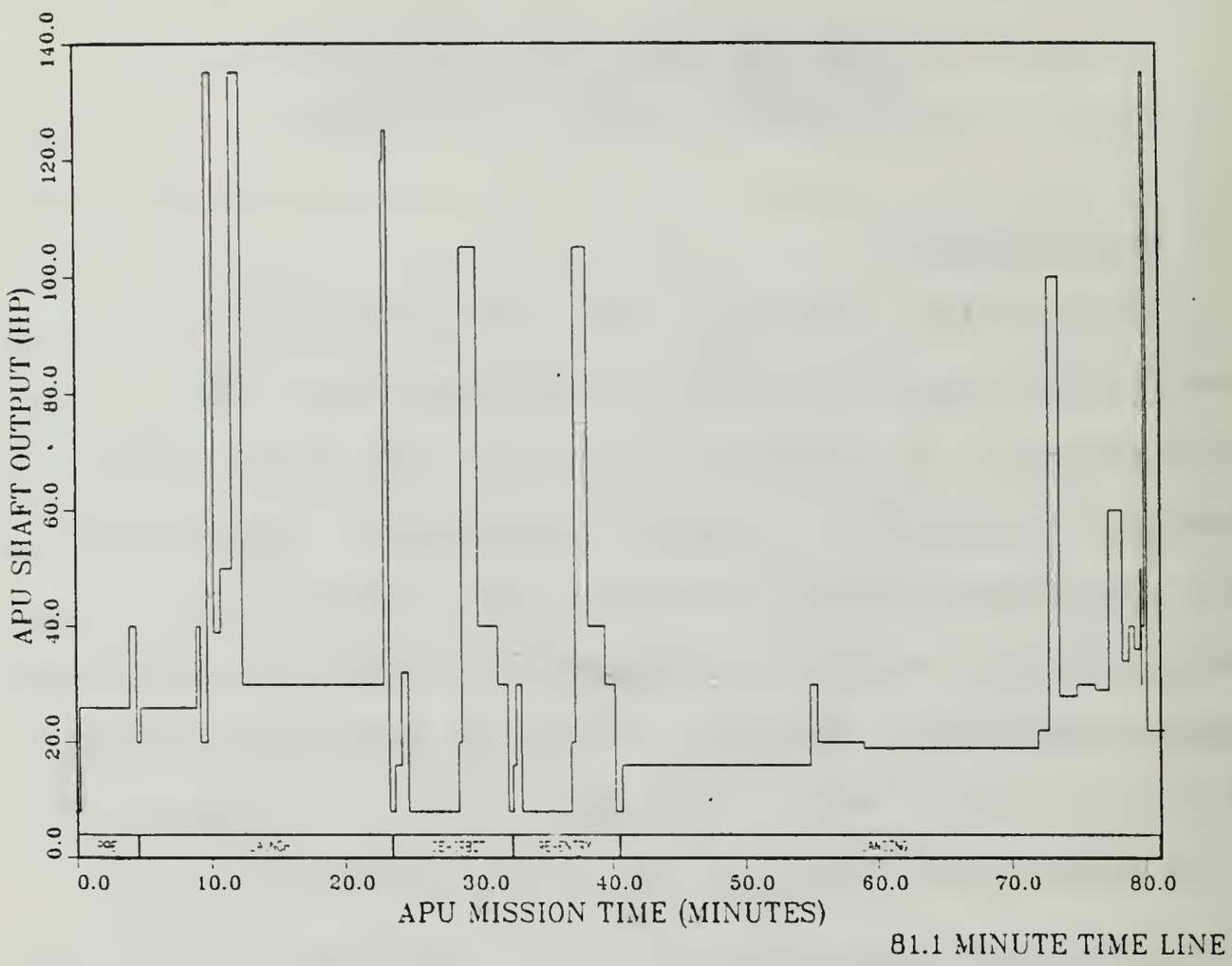


Figure 2.1 APU Mission Duty Cycle
(Baseline 81.1 Minute Time Line)

In the pre-launch phase APU loading varies minimally from 8.0 to 40.0 horsepower according to hydraulic requirements during the phase. Because the APU is designed to operate at 72,000 RPM's (plus or minus eight percent) over its entire range of output shaft horsepower, its steady state and dynamic vibrational characteristics vary little during the pre-launch phase. This is due to the minimal hydraulic loading which characterizes this phase.

APU output shaft horsepower is delivered to the hydraulic pump at a nominal 3800 RPM's through a two stage reduction gear. Although balanced to within exceedingly fine tolerances dictated by extremely fast rotational RPM's the APU is nevertheless characterized by a specific vibrational signature. We shall examine this signature in some detail because a careful understanding of its nature is crucial to the development of a filter dedicated to its detection.

In Appendix A there is included a graphical summary of the results of hotfire testing of one APU installed in the Shuttle Orbiter. These are Power Spectral Density (PSD) plots of accelerometers mounted along the x-axis (D0280A) and z-axis (D0281A) of this particular APU for various levels of loading. Also shown for reference is a PSD plot of the background noise prior to APU ignition. Relative to the plots of APU vibration we see that in each case the background noise is no less than two orders of magnitude lower than the PSD peaks of data obtained from the loaded APU.

The results of this test reveal a very particular vibrational signature for the APU. It should be noted that there are consistently repeating peaks at 600 Hz and two harmonics above this value at 1200 Hz and 1800 Hz. The bandwidth of the filter employed in this investigation had a 3 dB rolloff at 2300 Hz. This obviated a close examination

of higher spectral components although an expected 2400 Hz peak is present in several plots despite the attenuation.

The significance of these plots is in the consistency of the component spectral elements despite APU loading. Although the relative and absolute magnitudes of the fundamental 600 Hz peak and its harmonic constituents vary somewhat over the range of loading displayed the spectral location of each remains fixed. This shift in magnitudes is of some concern to mechanical engineers as it has been shown that failures in some rotating machinery have occurred when vibration signatures have deviated in this manner. But our concern is with the consistency of the location of the spectral peaks as this confirms the RPM stability of the APU over expected levels of loading.

We thus have a basis for further investigation of the acoustic environment in the Shuttle payload bay. We know that there is a specific vibrational signature which accompanies the normal functioning of the APU and we may expect that this vibration will translate to an acoustic signature in the Shuttle pre-launch environment. We now proceed to investigate this proposition with an examination of the acoustic environment present in the Shuttle payload bay prior to launch.

C. STS PAYLOAD BAY ACOUSTIC ENVIRONMENT MEASUREMENT INSTRUMENTATION

Data instrumentation recorders were flown on all early Shuttle flights to measure the acoustic environment present during the launch and landing phases of these missions. These experiments were conducted under the auspices of the NASA DATE (Dynamic, Acoustic and Thermal Environments) Working Group which has as its mission the development of improved methods for predicting all aspects of STS (Shuttle Transportation System) payload environments. Pursuant to this study we shall make use of data collected during each of the following three Shuttle missions:

-- STS-2 (conducted November 12-17, 1981)

-- STS-3 (conducted March 22-27, 1982)

-- STS-4 (conducted June 27 - July 4, 1982)

In each of these missions it is only the data which was recorded just prior to the launch which is of any significance for our purposes here.

Of particular interest to us is the acoustic data recorded at three specific locations in the Shuttle payload bay on each of the three flights listed above. On each flight sixteen selected sensors comprised the STS Development Flight Instrumentation (DFI) system. Of these sixteen sensors, four internal microphones were used to measure the acoustic environment present in the payload bay

at launch. Rounding out the array of acoustic sensors were three additional microphones mounted externally. All microphones were manufactured by Gulton Industries with only minor alterations differentiating the three external sensors from the four internal ones. The relative location of each of these seven acoustic sensors is shown in Figure 2-2.

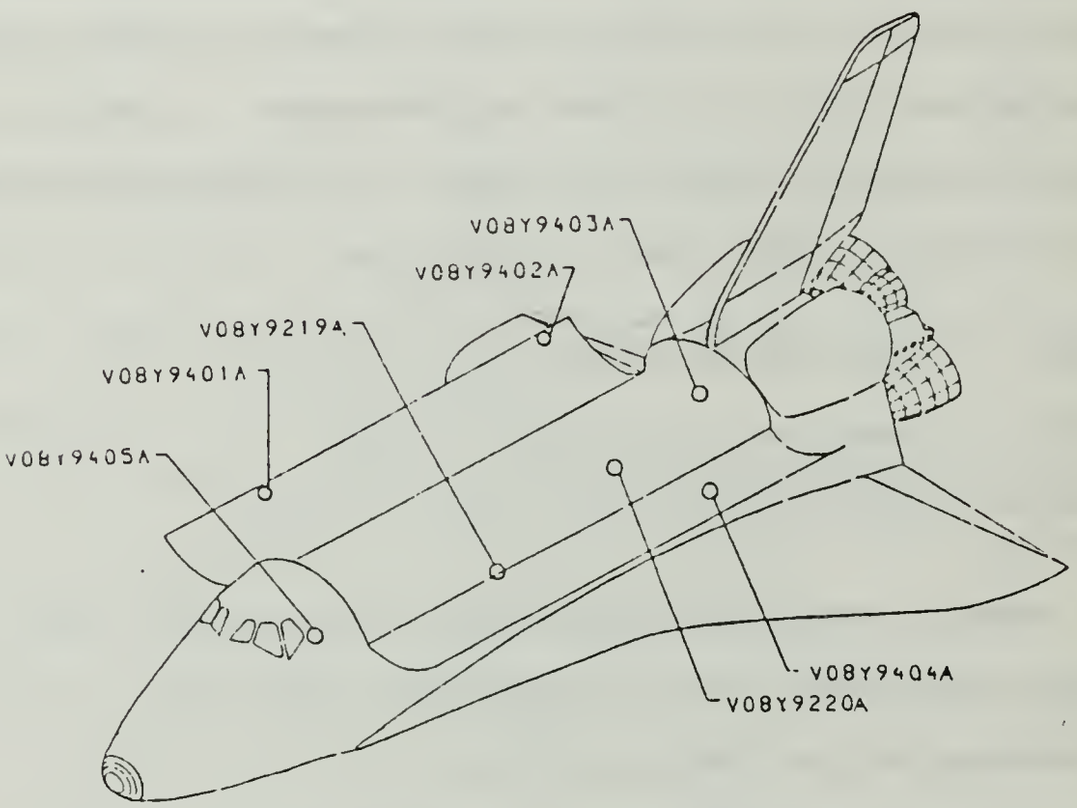


Figure 2.2 DFI Acoustic Measurement Locations

Sensor location designators, dynamic range and frequency response of each microphone are shown in Table 2-1 on the following page [Refs. 3 and 4].

Table 2.1

DEVELOPMENT FLIGHT INSTRUMENTATION (DFI)
ACOUSTIC MEASUREMENT INFORMATION

Microphone Number	Range (dB)	Cargo Bay Location	Orbiter Station Location X	Y	Z
V08Y9219A	110-157	Internal	863	-100	381
V08Y9220A	110-157	Internal	1190	0	427
V08Y9401A	130-177	External	639	3.5	500
V08Y9402A	130-177	External	1281	4.2	500
V08Y9403A	110-157	Internal	1306	12.0	400
V08Y9404A	130-177	External	1296	0	300
V08Y9405A	110-157	Internal	640	4	423

Note: The frequency response of all microphones is wide-band, 20 Hz to 8 kHz.

D. STS PAYLOAD BAY PRE-LAUNCH ACOUSTIC ENVIRONMENT

As indicated in the previous section we are principally interested in the response of three internal microphones to the Shuttle payload bay environment. The intent is to examine the environment prior to APU start-up to determine the noise background present before the APU acoustic signature is impressed upon this background. Then microphone responses will be examined after APU start-up to reveal the APU acoustic signature over the noise background. Based upon our previous examination of the APU vibrational signature evident in equipment testing we expect to see that spectral peaks at the fundamental and harmonic frequencies of 600 Hz will become clear at the time of APU start-up.

Appendix B is a summary of PSD plots of the acoustic environment present in the Shuttle payload bay at three different locations for the three missions listed in the previous section. As noted in Figure 2.2 the locations of the three microphones are dispersed throughout the payload bay to reveal variations in the acoustic signature from one location to another. Microphone V08Y9405A (which shall be referred to as 9405 for simplicity) was positioned near the forward bulkhead of the payload bay, furthest removed from the APU. Microphone V08Y9219A (9219) was located low and amidships, and microphone V08Y9403A (9403) was located at the after bulkhead, closest of the three to the APU. All three were identically configured to respond to a dynamic range of 110-157 dB.

The payload bay acoustic data is presented in pairs of plots with the first of the pair showing the signature before the APU is turned on and the second showing the signature approximately thirty seconds later after APU start-up. This data was obtained from an analog magnetic tape copy of the original data recording which was made available from the DATE group at Aerospace Corporation headquarters in Los Angeles.

There is doubtless some extraneous noise introduced in the process by which the original tape was copied to yield the tape available at Aerospace. Furthermore, the generation of the Aerospace copies are not known. However

it is known that measures were taken to minimize the error introduced in the dubbing process. On most copies a Tape Speed Compensation (TSC) process was employed which ensures that the tape transport travels at the same speed as when originally recorded. That this process is indispensable is realized upon examination of the PSD's obtained for microphones 9405 and 9219 on STS-3. In these two cases alone TSC was not employed due to operator error at a previous generation. The noise floor thus generated is seen to be an order of magnitude greater than in other plots and is severe enough to mask the desired information. In all other tape copies TSC was employed.

(Another generation of tape copies was recorded for use at the Naval Postgraduate School in the development of the experiment described earlier. This recording process was accomplished using a Hewlett-Packard Model 3964A Instrumentation Data Recorder in an FM mode to allow recording of analog data from 0 to 8 kHz. The HP recorder also employs a tape speed servo control which also ensures tape speed accuracy. PSD plots were obtained in all cases for the dubbed tape to verify the accuracy of this latest dubbing routine. In each case the PSD of the dubbed data was confirmed to be a good reproduction of the "original" except for the region below 200 Hz. In this region the reproduction was not as good as in the region above 200 Hz

but for our purposes this is an inconsequential shortcoming.)

An examination of the PSD's of the remaining sets of data not corrupted by the lack of TSC reveals the expected behavior. In all cases the noise background remains essentially unchanged from the first of the plot pair to the second except for the effect of the impressed APU signature clearly evident in the second. In every plot performed after APU start-up the characteristic 600 Hz peak is evident to some degree and in many instances we also see the harmonic components at 1200 and 1800 Hz. This is especially true for those plots reflecting the environment surrounding microphone 9403 which is located only a bulkhead removed from the APU's themselves. However, of some concern is the fact that the 600 Hz peak is least evident for microphone 9405 which is at the forward bulkhead furthest removed from the APU's and where it is expected that the GAS canister will be placed for the upcoming mission.

The salient points are therefore as follows:

1. The APU does provide a specific signature which becomes clearly evident in the audio spectrum of the Shuttle payload bay pre-launch acoustic environment. There are well-defined spectral peaks at a fundamental frequency of 600 Hz and at integral harmonics thereof.

2. The magnitude of the APU signature is variable in the payload bay depending upon the location of the sensor used to detect it. A sensor placed closer to the after bulkhead will be more apt to respond to the APU signature in a manner which will facilitate matched filter performance, but the signature is evident throughout the payload bay.

Therefore our task is to develop a matched filter which will detect the presence of a spectral peak at 600 Hz and/or at coincident harmonic multiples of 600 Hz. We will attempt to realize the needed performance emphasizing only the most evident 600 Hz peak. Based upon the results of this design, will then know if we must consider the additional impact of the less evident harmonic elements.

Our desire is to implement the scheme which provides the simplest, smallest, most reliable, and least power consuming design which will ensure APU detection prior to launch. Therefore detection of all higher order harmonic components is less important than the development of a filter which accomplishes the primary goal. We shall examine both analog and digital methods for implementing our matched filter and compare the effectiveness of the various designs before selecting the device which will orbit the Earth.

III. ANALOG FILTER DESIGN THEORY AND IMPLEMENTATION

The traditional method of filter implementation in electronic circuitry was for a long time characterized by an implementation of passive and discrete resistive, inductive and capacitive components tuned to respond to the desired frequency components of the input signal. In the earlier years of circuit design this procedure involved a lengthy process of theoretical development and precise component selection. This was often a tedious process involving component trial and substitution.

Analog filter design took a great leap forward with the advent of integrated operational amplifiers (op-amps) and later, integrated circuit (IC) technology. Now for instance, using an integrated circuit such as the National Semiconductor AF100 Universal Active Filter as a design basis, it is possible for a circuit designer to construct a precise analog filter circuit with a surprising economy of effort. We will use the Biquad Elliptic Filter design which forms the basis of the AF100 to implement the analog designs we shall develop herein.

A. PRACTICAL DEVELOPMENT OF AN ANALOG BAND-PASS FILTER

We will begin our development of a tuned filter design by examining an analog IC implementation of a band-pass

filter with a center frequency of 600 Hz. (We could expand this band coverage to include the two additional center frequencies of 1200 Hz and 1800 Hz if it proves that such a design modification is necessary.) In this development we shall choose to build an Elliptic (or Cauer) filter which exhibits a much steeper roll-off outside the passband over a Butterworth or Chebyshev design of equivalent order. The disadvantage of ripple in the passband, which characterizes Elliptic filters, will cause minimal impact and will not be a factor in the realization of our goal. In fact we can allow the ripple in the passband to be relatively high because our intent is not to pass a faithful representation of the APU signature but only to detect its presence. Thus our goal is to construct a band-pass filter with a high quality factor and narrow passband. This corresponds to a steep roll-off out of the passband.

The method which will be employed to realize this band-pass filter will be to describe the characteristics of the desired analog band-pass design and, using a low-pass-to-band-pass transformation, solve for the form of the corresponding low-pass prototype using transform relations. This will allow us to determine the necessary order of the low-pass design which can subsequently be transformed into the required band-pass filter.

If we again examine the PSD plots of the APU noise above the background for the 600 Hz component we can generalize a

desired filter transfer function to approximate this response. Let us choose the 3 dB points of the band-pass filter (centered at 600 Hz) to be at 575 and 625 Hz. We will allow the pass-band ripple width (PRW) to be as much as 2 dB within the pass-band. Let us furthermore require the stop-band attenuation on either side of the pass-band to be down at least 30 dB at 500 and 700 Hz. This represents a very steep roll-off characteristic and suggests the use of an Elliptic filter design for that reason. In fact, to achieve this degree of roll-off in a low-pass design would require a model prototype of order 6. The equivalent Chebyshev design would require a minimum order of 14, while the Butterworth low-pass filter equivalent order would be at least 63. Clearly the Elliptic design is our only viable alternative.

The low-pass to band-pass transformation for analog filters results in a transfer function which raises the order of the low-pass equivalent by a factor of two. Therefore if we design a band-pass filter it will necessarily consist of a number of second-order stages in the final implementation.

1. Low-Pass to Band-Pass Frequency Transformation

In order to develop the transfer function of an appropriate analog band-pass filter we must begin with the transfer function of the corresponding analog low-pass filter which may be transformed into the desired band-pass

filter by a frequency transformation. However, the characteristics of our final filter are known in band-pass form. Thus we must deduce the analog low-pass design from the band-pass characteristics and then apply the analog transformation to the low-pass prototype to realize our goal. This development is a combination of procedures described in Chen [Ref. 5] and Johnson [Ref. 6].

We wish to design an analog band-pass filter having the following characteristics

f'_{p_2} = 575 Hz

f'_{q_2} = 500 Hz

f'_{p_1} = 625 Hz

f'_{q_1} = 700 Hz

PRW = 2.0 dB (allowable ripple in the passband)

MSL = 30 dB (minimum attenuation in stop-band)

In this analog development the prime frequencies refer to the band-pass function and the unprimed frequencies to the low-pass function. Figure 3.1 is a graphical depiction of the relationship between the transfer function transform pair. The negative axis frequencies arise from the mathematics of this development. However, we shall only be functionally concerned with the positive axis transformation. We are also considering the low-pass function to be normalized (ω = 1).

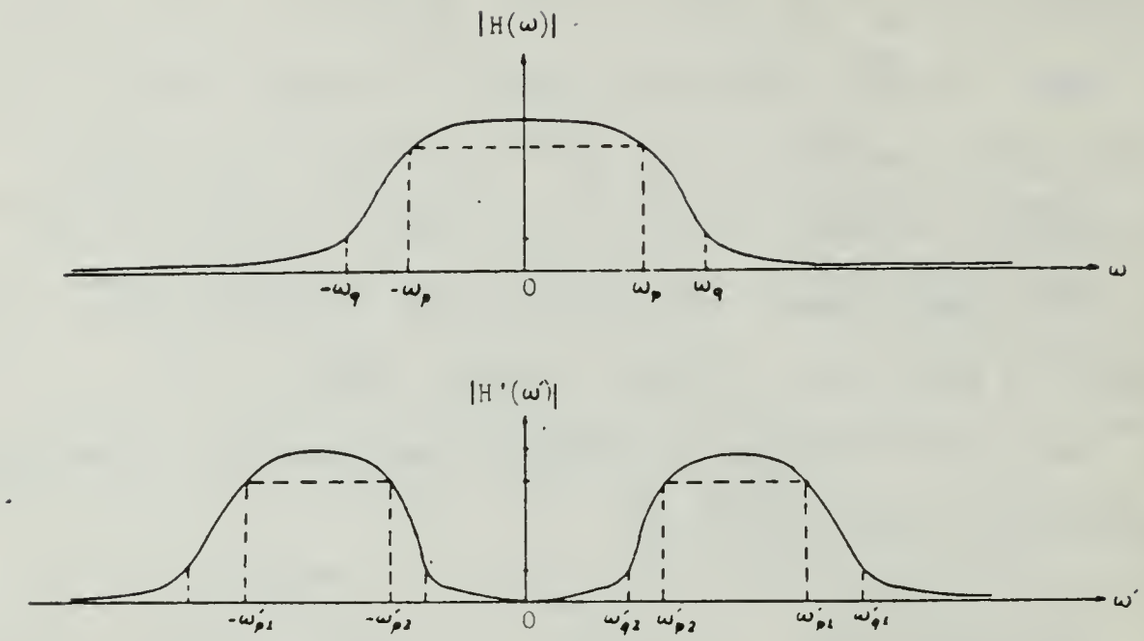


Figure 3.1 A Low-Pass to Band-Pass Frequency Transformation in the Analog Domain

A suitable transformation must therefore accomplish the transform relations detailed in Table 3.1 [Ref. 5].

Table 3.1

LOW-PASS TO BAND-PASS TRANSFORM PAIRS

Low-Pass Function	Band-Pass Function
$\omega = \infty$	$\omega' = \infty$
$\omega = 1$	$\omega' = \omega'_{p1}$
$\omega = -1$	$\omega' = \omega'_{p2}$
$\omega = -\infty$	$\omega' = 0$

As developed in Chen [Ref. 5: p. 235], the analog frequency transformation which will accomplish a low-pass to band-pass frequency transformation [H(s) ==> H(s')] is given by the following relation.

$$s = \frac{s'^2 + \omega'_{p1}\omega'_{p2}}{s'(\omega'_{p1} + \omega'_{p2})}$$

or if we substitute the above values

$$s = \frac{s'^2 + 4\pi^2 \cdot (3.59375 \times 10^5)}{s' \cdot 2\pi \cdot 50}$$

or, using $s = j\omega$

$$\omega = - \frac{4\pi^2 \cdot (3.59375 \times 10^5) - \omega'^2}{\omega' \cdot 2\pi \cdot 50}$$

Making the following substitutions into the preceding equation yields

$$\omega'_{q1} = 2\pi \cdot 500$$

$$\Longrightarrow \omega_{q1} = -4.3750000$$

and

$$\omega'_{q2} = 2\pi \cdot 700$$

$$\Longrightarrow \omega_{q2} = -3.7321429$$

Therefore we are left with the need to design an analog low-pass filter with

$\omega_c = 1$ rad/sec, and

$\omega_q = 3.7321429$ rad/sec

This leaves us with a normalized low-pass transition width (TW_{lp}) of

$$TW_{lp} = \omega_q - \omega_c$$
$$= 2.7321429$$

We now have enough information to enter the tables in Johnson [Ref. 6] to obtain the data shown in Table 3.2.

Table 3.2

ELLIPTIC LOW-PASS FILTER DATA
(for the band-pass filter low-pass prototype)

A	B	C	WZ	WM	KM
21.16400	0.787152	0.842554	4.600435	0.715610	1.258925

2. 600 Hz Elliptic Band-Pass Filter Design

In the case of elliptic band-pass filters the transfer function may be factored into the product of second-order functions. The two factors arising from each second-order low-pass stage have the forms [Ref 6: p. 100]

$$\left[\frac{V_2}{V_1}\right]_1 = \frac{K_1\sqrt{C/A}(s^2 + A_1\omega_0^2)}{s^2 + (D\omega_0/E)s + D^2\omega_0^2} \qquad 3.1$$

and

$$\left[\frac{V_2}{V_1}\right]_2 = \frac{K_2\sqrt{C/A}(s^2 + \omega_0^2/A_1)}{s^2 + (\omega_0/DE)s + \omega_0^2/D^2)} \qquad 3.2$$

where

$$E = \frac{1}{B}\sqrt{\frac{C + 4Q^2 + \sqrt{(C + 4Q^2)^2 - (2BQ)^2}}{2}}$$

$$D = \frac{1}{2}\left[\frac{BE}{Q} + \sqrt{\frac{(BE)^2}{(Q)^2} - 4}\right]$$

and

$$A_1 = 1 + \frac{1}{2Q^2}(A + \sqrt{A^2 + 4AQ^2})$$

and $Q = f_0/BW = 600/50 = 12$. The coefficients A, B and C are those of the normalized low-pass function given in Table 3.2 above, and K_1 and K_2 are related to the stage gain K by $K = K_1K_2$.

Equations 3.1 and 3.2 above are of the general form

$$\frac{V_2}{V_1} = \frac{\rho(s^2 + \alpha\omega_0^2)}{s^2 + \beta\omega_0 s + \gamma\omega_0^2} \qquad 3.3$$

which is identical to the form of the low-pass transfer function, except for the replacement of ω_0 by the corresponding low-pass term ω_c.

Our analog band-pass filter will have two stages of the form given by Eq. 3.3. Comparing Eq. 3.3 with Eq. 3.1 and Eq. 3.2 reveals the following transfer function coefficients of the band-pass filter stages [Ref 6: p. 118]:

1) First stage

$$\rho = K_1 \sqrt{C/A}$$

$$\alpha = A_1$$

$$\beta = D/E$$

$$\gamma = D^2$$

2) Second stage

$$\rho = K_2 \sqrt{C/A}$$

$$\alpha = 1/A_1$$

$$\beta = 1/DE$$

$$\gamma = 1/D^2$$

In the FORTRAN program ABPDBP (included as Appendix C) many of the calculations which are indicated in this thesis development will be performed. In Section 1 of this program we begin with desired filter parameters and tabulated values which correspond to the filter we wish to build. We then calculate several derived parameters from these initial values. Next we perform further calculations (which shall be developed shortly) which yield values for filter resistors and capacitors.

In Section 2 of ABPDBP we use the two sets of filter parameters which correspond to each of the filter stages indicated by Eq 3.3 to arrive at the overall transfer function. ABPDBP describes the corrections which must be made to provide for pre-warping of frequencies preparatory to a digital transformation (which shall be discussed in Chapter 4) but these changes can be ignored in this analog discussion. Thus we will use f_o = 600 Hz (which implies the use of W0, not W0DIG) in the program calculations. The fourth order analog filter transfer function which results from this development is given in Equation 3.4.

$$H(s) = \frac{a_0 s^4 + a_1 s^3 + a_2 s^2 + a_3 s + a_4}{b_0 s^4 + b_1 s^3 + b_2 s^2 + b_3 s + b_4} \qquad 3.4$$

The values of the coefficients in this analog transfer function representation are given in Table 3.3 which follows.

Table 3.3

ANALOG ELLIPTIC BPF FOURTH ORDER COEFFICIENTS
(Normalized coefficients with gain 0.03981)

Coefficient	Value
a_0	1.0000
a_1	0.0
a_2	2.9587×10^7
a_3	0.0
a_4	1.8991×10^{14}
b_0	1.0000
b_1	2.4351×10^2
b_2	2.7642×10^7
b_3	3.3558×10^9
b_4	1.8991×10^{14}

In Section 2A of the program ABPDBP the poles and zeros of the analog transfer function are then calculated to demonstrate the stability of the filter design. The values for the poles and zeros may be observed in the output of ABPDBP and are reproduced graphically in Figure 3.2 on the following page. The poles of the filter lie within the left

half of the s-plane and this confirms the stability of our design.

3. Analog Band-Pass Filter Simulation

Now that we have developed the transfer function which describes the desired analog band-pass filter we can use this function to simulate the active operation of the

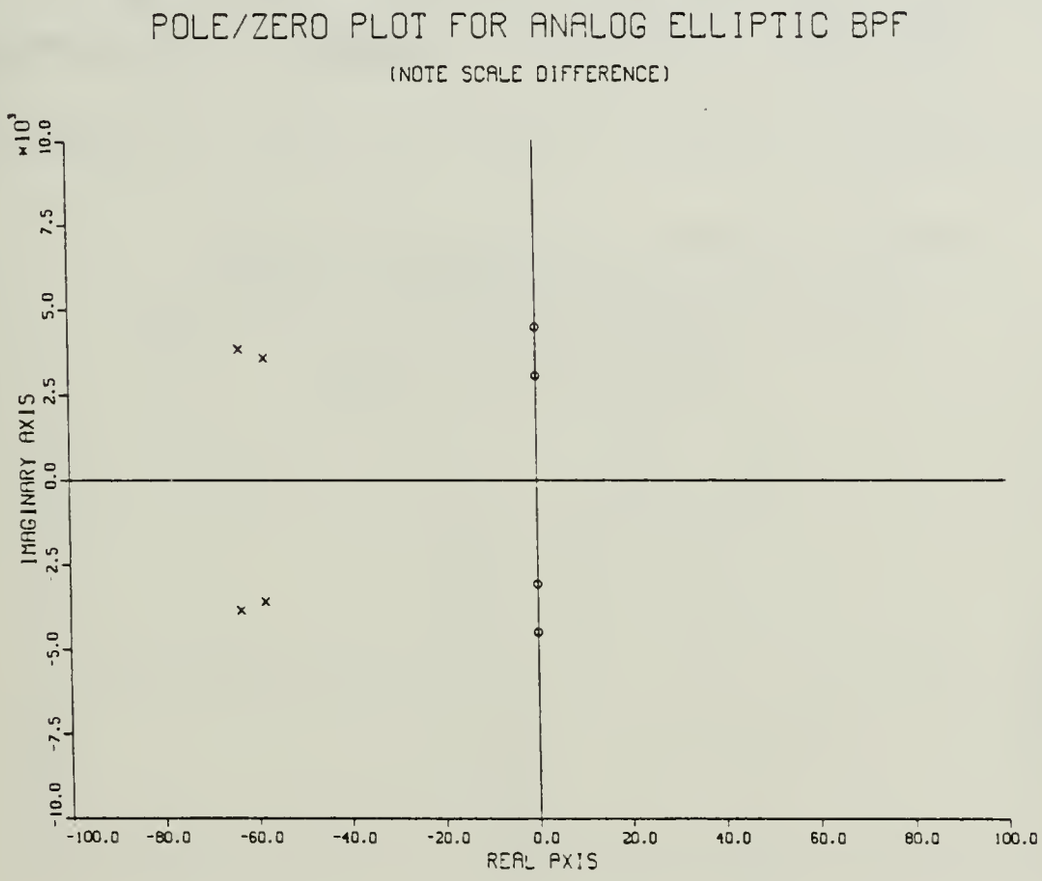


Figure 3.2 Pole/Zero Plot for the Analog Elliptic Band-Pass Filter

filter. The FORTRAN program ABPFR (which is included as Appendix D to this thesis) is used to examine this particular band-pass filter simulation. Figures 3.3, 3.4

and 3.5 which follow are the results of this computer simulation of the filter response for the device we have just designed. The range of frequencies of the simulated computer input is DC to 1 kHz. The simulated amplitude is constant over the range of input frequencies.

In Figure 3.3 we see the amplitude response which is near zero at all but the passband frequencies around 600 Hz. Between 500 and 700 Hz we confirm the desired filter response. The center frequency is located at 600 Hz and

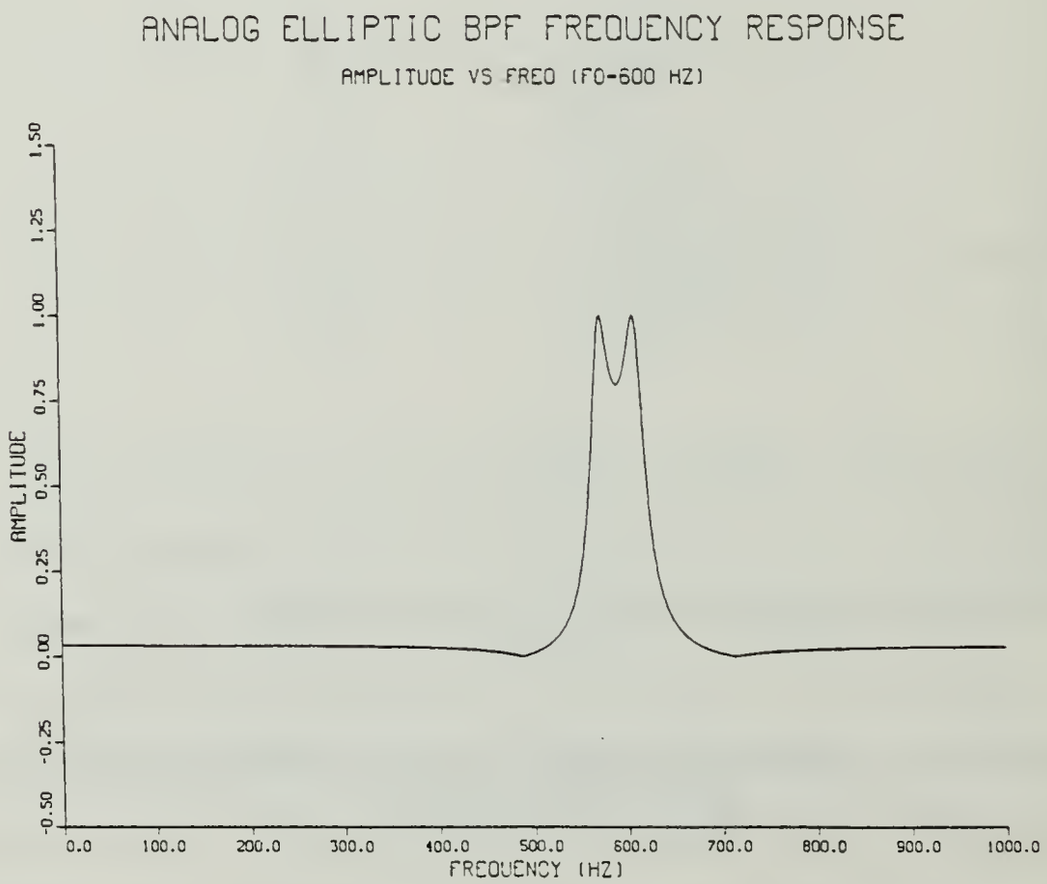


Figure 3.3 Analog Elliptic BPF Frequency Response (Computer Simulated Amplitude Response)

there is a significant minimum at the center frequency due to the effect of the passband ripple of 2.0 dB. Furthermore we observe a half power point (3.0 dB down point) at about 575 Hz and 625 Hz as specified in our design.

In Figure 3.4 we again observe a computer simulation of the amplitude response of the filter, this time measured in decibels. The marked presence of notches at about 500 Hz and 700 Hz is obvious, and the 30 dB minimum loss in the stopband is also confirmed. We also have graphical confirmation of the 2.0 dB ripple width in the passband.

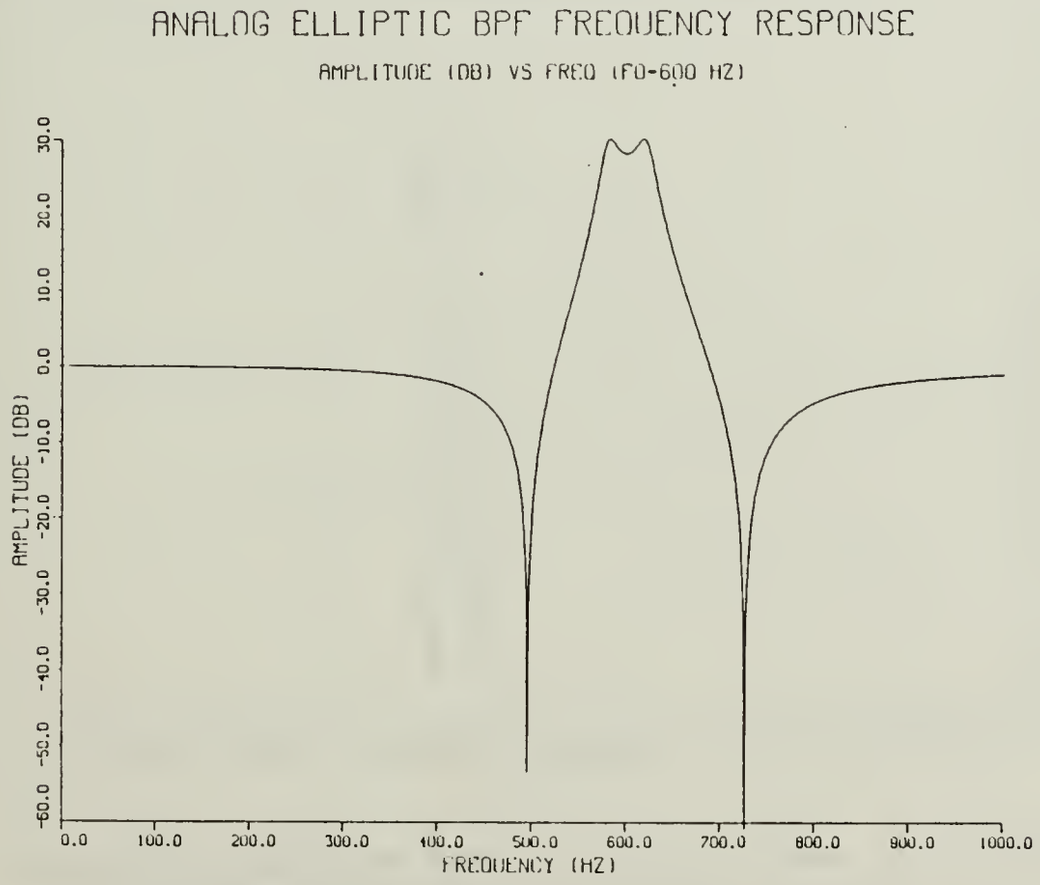


Figure 3.4 Analog Elliptic BPF Frequency Response (Computer Simulated Amplitude Response in dB)

Finally, in Figure 3.5 we view the computer simulation of the analog filter phase response. Although our application is not phase dependent (due to the fact that it is the presence alone of the 600 Hz element which is of concern to our circuit--not its accurate transmission), we do confirm the significant effect upon phase for our elliptic filter in the passband between 500 Hz and 700 Hz. Between these two frequencies we observe a 360 degree shift in phase.

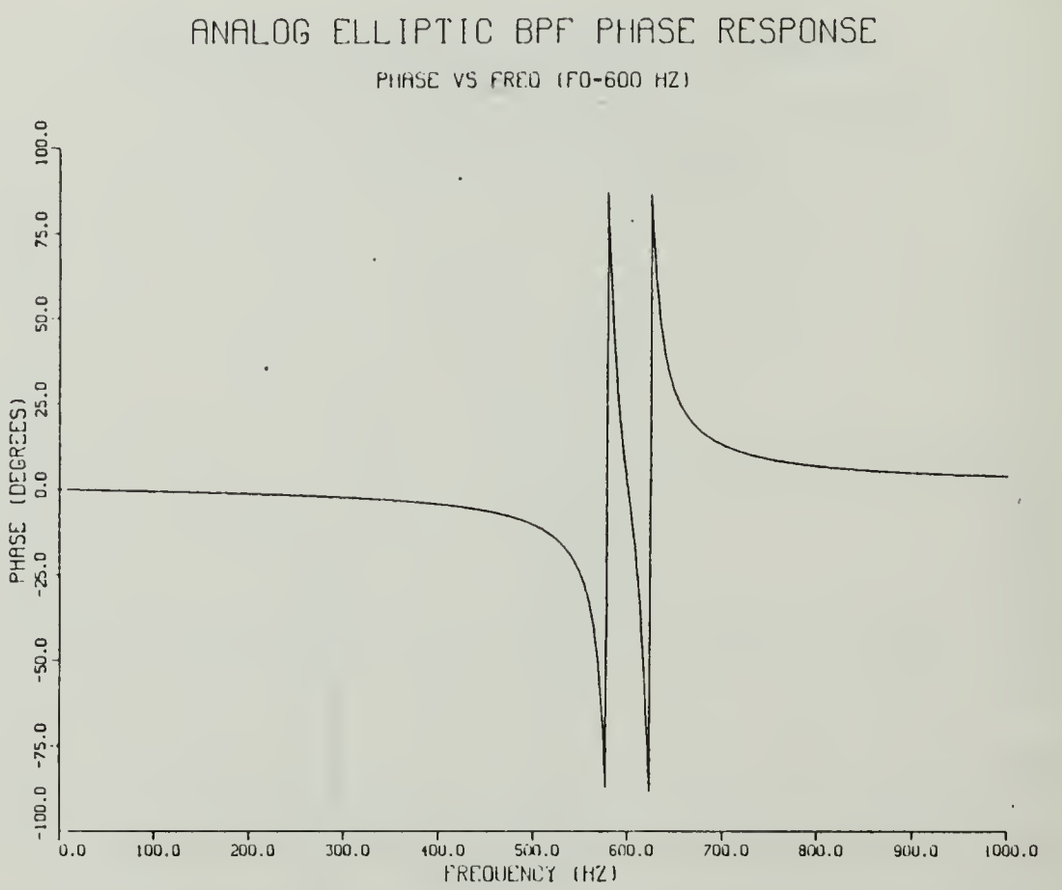


Figure 3.5 Analog Elliptic BPF Frequency Response (Computer Simulated Phase Response)

B. HARDWARE IMPLEMENTATION

1. Biquad Analog Band-Pass Elliptic Filter

There are many ways to perform a hardware implementation of the analog band-pass transfer function we have just developed. One relatively easy method employs the use of op-amps as the active filter component. We shall use a Biquad op-amp filter implementation which exhibits good stability and ease of tuning. Additionally, implementation is made simpler by the use of a 74124 quad op-amp microchip which allows a single chip per second-order stage. The generalized circuit diagram for a second-order stage of a Biquad filter is shown in Figure 3.6 [Ref. 6: p. 127].

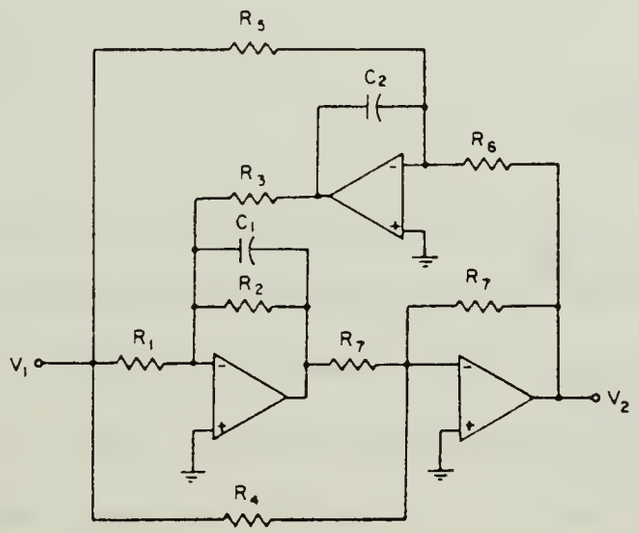


Figure 3.6 Biquad Elliptic Filter Circuit

Component resistor and capacitor values for the Biquad filter depend upon the low-pass normalized

coefficients and the derived complements we have already evaluated. We begin our development by choosing a standard value for C_1 (given roughly by $10/f_0$ μF) and then proceeding to calculate elemental values. In the equations which follow the values for C_2 and R_7 are arbitrary within limits, and are chosen to minimize the spread of resistance values. We pick $C_2=C_1$ and $R_7 \approx 1/(\omega_0 C_1)$. A_1, E and D are as given previously.

The first stage values are thus [Ref. 6: p. 126]:

$$R_1 = \frac{E\sqrt{A/C}}{KD\omega_0 C_1}$$

$$R_2 = KR_1\sqrt{C/A}$$

$$R_3 = \frac{1}{D\omega_0 C_1}$$

$$R_4 = \frac{A}{C}\sqrt{R_7/K}$$

$$R_5 = \frac{D\sqrt{A/C}}{KA_1\omega_0 C_2}$$

$$R_6 = \frac{C_1 R_3}{C_2}$$

Similarly, the second stage values are given by

$$R_1 = \frac{DE\sqrt{A/C}}{K\omega_0 C_1}$$

$$R_2 = KR_1\sqrt{C/A}$$

$$R_3 = \frac{D}{\omega_0 C_1}$$

$$R_4 = \frac{A}{C}\sqrt{R_7/K}$$

$$R_5 = \frac{A_1\sqrt{A/C}}{KD\omega_0 C_2}$$

$$R_6 = \frac{C_1 R_3}{C_2}$$

In Section 1 of the program ABPDBP (introduced as Appendix C) we calculate the resistor values for the Biquad band-pass circuit just discussed. The resulting component values which were thus derived are shown in the appendix and are also included here as Table 3.4. These are computed values for resistance and capacitance. In fact the circuit is constructed by selecting standard values as close as

possible to these values and then tuning the circuit for the desired performance. Tuning is accomplished in each stage by adjusting R_4 to set the notch frequency f_z, R_3 to set the center frequency f_0, R_2 to set Q, and R_1 or R_5 to set the gain.

Table 3.4

BIQUAD ELLIPTIC BAND-PASS FILTER COMPONENT VALUES

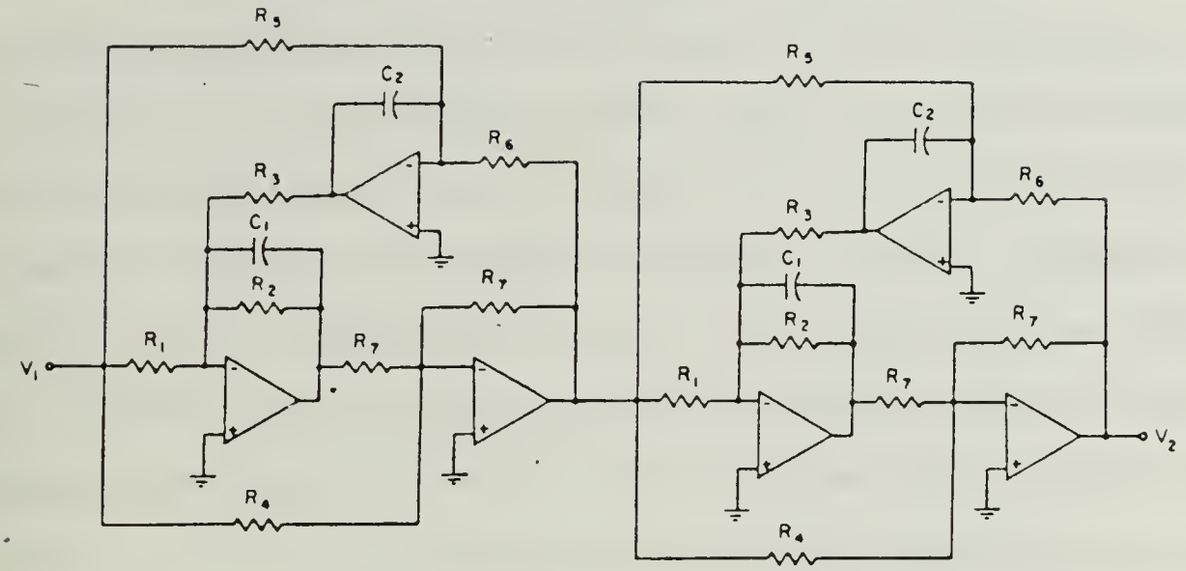
1st Stage		2nd Stage	
Component	Value	Component	Value
C_1	.00996	C_1	.01030
C_2	.00995	C_2	.01034
R_7	26.7	R_7	26.7
R_6	25.7	R_6	26.5
R_5	10.5	R_5	20.2
R_4	14.8	R_4	14.8
R_3	25.7	R_3	26.6
R_2	785.	R_2	813.
R_1	437.	R_1	452.

Note: Capacitor values are μF, resistor values are kΩ.

The resulting schematic for the fourth-order Biquad elliptic band-pass filter is shown in Figure 3.7.

2. Follow-Up Logic Circuitry

When the band-pass filter is implemented the effect is to produce a response which narrowly limits the passband to within a few tens of hertz about the center frequency of 600 Hz. Still, the output of this filter will be of



First Stage Second Stage

Figure 3.7 Biquad Elliptic Band-Pass Filter Circuit

sinusoidal nature and inadequate to drive a microprocessor interrupt designed to accommodate TTL logic levels. Thus we must include further circuitry into our design which will send a TTL compatible logical signal to the microprocessor when the APU signal is detected. The circuit which accomplishes this is shown in Figure 3.8.

The output from the filter stages is first sent to a linear amplifier constructed around a 741 op-amp. Because the APU signal is low voltage out of the microphone detector it is necessary to amplify the filter output prior to logical evaluation.

The decision of whether or not a 600 Hz component is present is the function of comparator and counter elements

which follow the amplifier. If the amplitude of the amplified filter output goes above the threshold set at the reference input of the LM311-based comparator then a pulse is developed for the duration that the input signal exceeds the threshold level. A negative-edge triggered 74121-based one-shot multivibrator follows the comparator. It is designed to send a one millisecond pulse to the counter stages which follow any time the comparator detects an input signal which exceeds the threshold level.

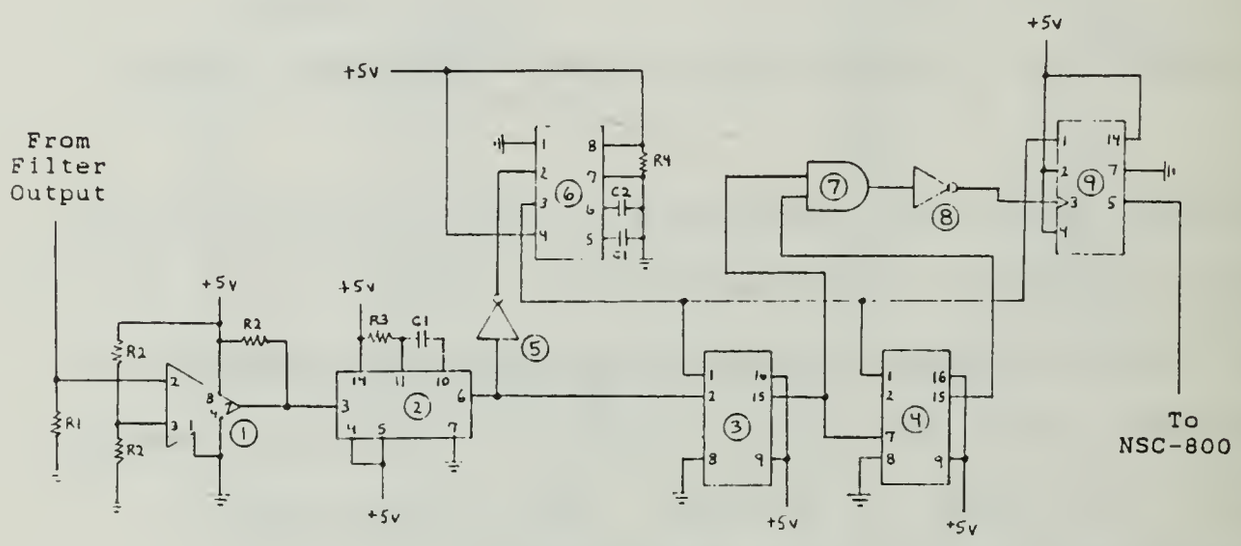


Figure 3.8 Follow-Up Pulse-Shaping Logic Circuitry

Microcircuits	Components
1: LM311 Op-Amp	R1: 100 kΩ
2: 74121 1 msec One-Shot	R2: 20 kΩ
3: 74161 4 Stage Counter	R3: 15 kΩ
4: 74161 4 Stage Counter	R4: 470 kΩ
5: 7404 Inverter	C1: 0.1 μF
6: 555 0.5 sec Timer	C2: 1.0 μF
7: 7432 AND Gate	
8: 7404 Inverter	
9: 7474 D-Type Flip-Flop	

The two 74161 counter stages which follow the multivibrator are designed to count up to 255 occasions of the threshold being exceeded in a .5 second period before the decision is made that a valid 600 Hz signal was detected. This is an arbitrary figure. The .5 second period is established by the 555 timer which is also fed by the one-shot multivibrator. If the counter stages do not sum to 255 within a .5 second period then the 555 resets the counter stages to zero and counting begins anew with the comparator. If the counter does reach 255 within the .5 second period then a latch is set for the remainder of the .5 second period. This TTL level signal is the one which provides the microprocessor interrupt indicating that the APU signal has been detected.

C. ANALOG BAND-PASS FILTER IMPLEMENTATION RESULTS

Figure 3.9 on the following page is a photograph of the actual frequency response of the analog elliptic band-pass filter we have developed. A sweep generator was used to input a ramped sinusoidal input comprised of a linear continuum of frequencies (generated by application of a skewed triangular input to a voltage controlled frequency oscillator) in the range of approximately 100-1000 Hz. Because the ramp generator does not exhibit an instantaneous return, the return also generates a down-frequency response albeit at a rate greater than that of the up-frequency ramp.

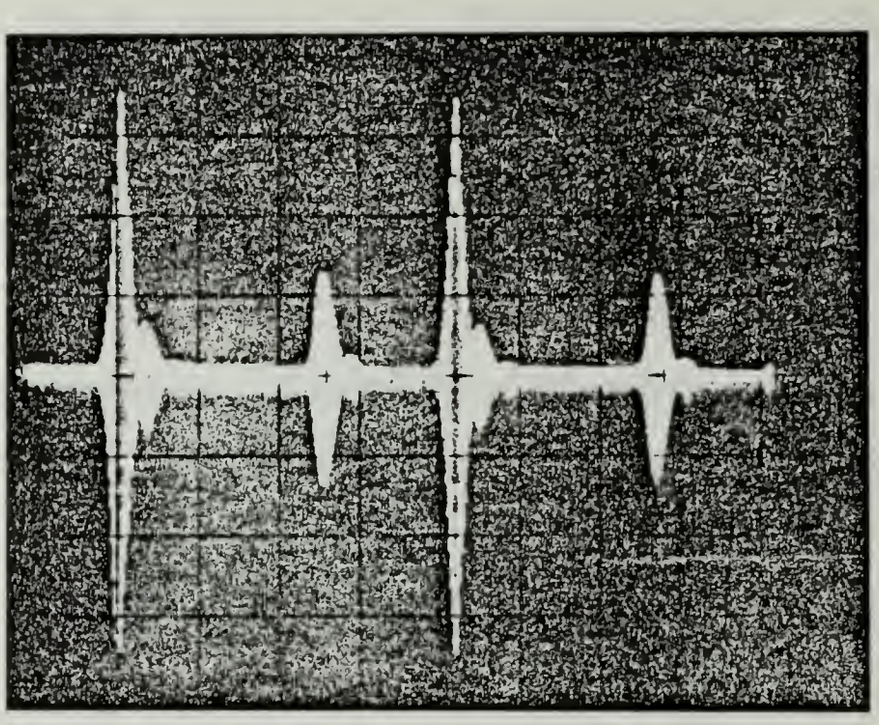
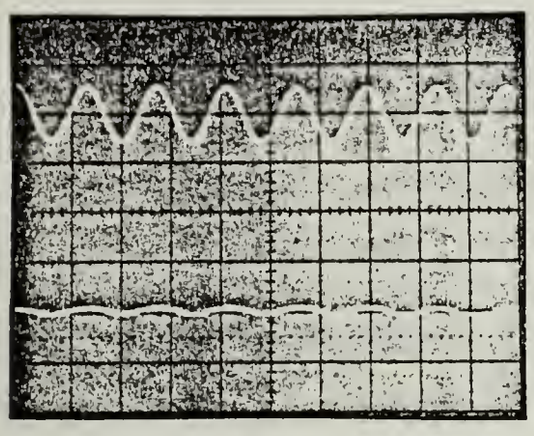


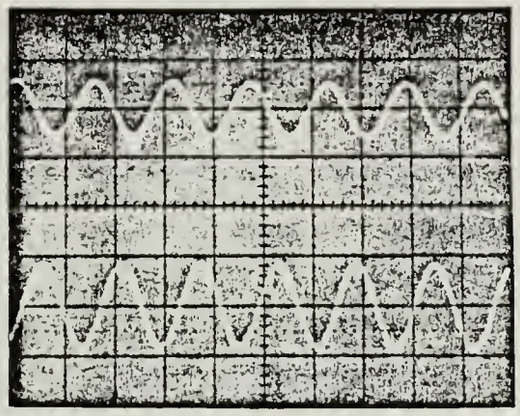
Figure 3.9 Analog Elliptic BPF Frequency Response
(Photograph of Actual Filter Amplitude Response
to a Ramped Sinusoidal Input)

This gives rise to the appearance of a double response which is noted in the figure. Actually we are observing a multiple reponse over successive up and (faster) down ramps of the input sinusoid. Thus we are able to observe graphical confirmation of the filter amplitude response predicted in the foregoing discussion.

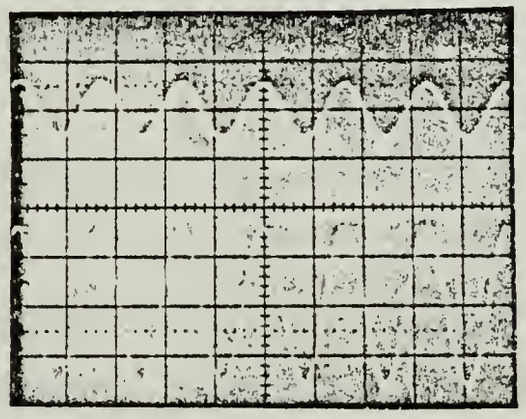
As expected we observe a very narrow filter bandpass response (with frequency limits we will look at more closely in the following paragraph). The curious extended response ("hump") at the upper end of the passband is due to the inexact placement of poles and zeroes accomplished by tuning of the filter response in the aforementioned manner.



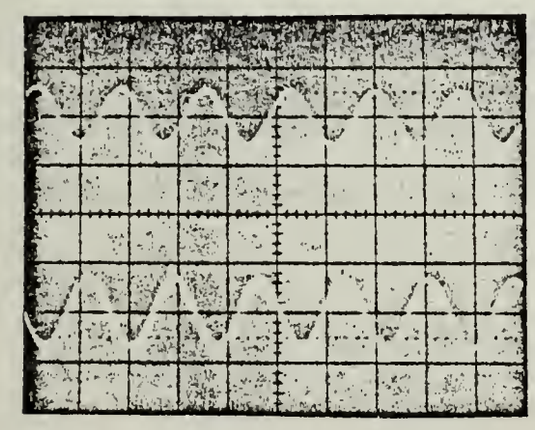
a) 500 Hz



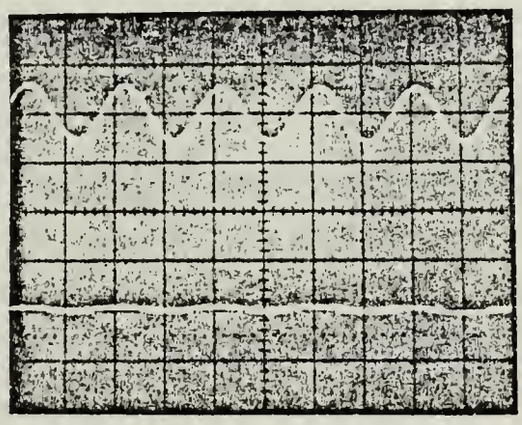
b) 575 Hz



c) 600 Hz



a) 625 Hz



b) 700 Hz

Figure 3.10 Analog Elliptic BPF Frequency Response
Upper trace (Input): 50 mV/div scale
Lower trace (Output): 1.0 V/div scale

In Figure 3.10 we examine this response more specifically for discrete frequencies in the range of 500 Hz to 700 Hz. Instead of applying a ramped sinusoid we input five discrete sinusoids while maintaining a constant amplitude. Thus we again observe the very narrow bandpass filter response at least within the limits presented here. We also confirm the rapid shift in phase of 180 degrees from the lower to the upper bound in agreement with theory. While this examination by itself does not confirm the desired filter response, it does so when considered with the results of the previous figure.

In Chapter 4 we will use the results of this analog filter implementation to develop an equivalent digital realization. To do this we will use common transformation techniques to arrive at a z-domain transfer function which we will then reduce to a difference equation. This format will then allow us to realize a digital elliptic filter by use of the INTEL 2920 Signal Processor. This hardware realization of the digital filter will be accomplished in Chapter 5.

IV. DIGITAL FILTER DESIGN

When designing an IIR digital filter for a specific application it is common practice to first develop an analog filter with appropriate characteristics as we did in the preceding chapter. Once the analog design is attained it is then possible to transform this analog filter into a digital filter with the desired passband characteristics.

There are several reasons why it is desirable to use this approach [Ref. 7: p. 5-7]. Of primary importance is the fact that the art of analog filter design is highly advanced. Consequently there are many techniques available for implementing specific designs. Because useful results can be achieved, following established analog design procedures presents advantages in the amount of effort which must be spent in the design phase.

Additionally, many useful analog design methods have relatively simple closed-form design formulas. This greatly facilitates the implementation of the corresponding digital filters.

Finally, in many applications it is of interest to use a digital filter to simulate the performance of an analog linear time-invariant filter.

There are many alternative methods for accomplishing a transformation of fixed filter characteristics. In each

case known analog filter characteristics in the frequency domain (the Laplacian "s" domain) are converted to similar characteristics in the digital "z" domain. Each of these techniques introduces a non-linearity into the resulting amplitude and phase characteristics of the original analog filter. If necessary to preserve the phase, equalizers may be employed to return the phase characteristic to a nearly linear behavior over the region of interest in the digital domain. In our case any phase distortion can be ignored because we are only interested in frequency detection and not accurate reproduction.

Generally speaking, when beginning with an analog low-pass design, we may proceed in a number of ways to arrive at a corresponding digital band-pass filter realization. For instance, we may first transform the low-pass filter to an analog band-pass design (as we did in Chapter 3 for the low-pass to band-pass transformation) and then employ an analog to digital transformation to yield the digital filter. Alternatively we may choose to employ the analog to digital transform on the low-pass filter and then apply a digital low-pass to digital band-pass transformation to realize our goal. Finally, it is also possible to combine these two-step routines into a single-step analog low-pass to digital band-pass direct transformation. These options are shown in Figure 4.1 [Ref. 5: p. 269].

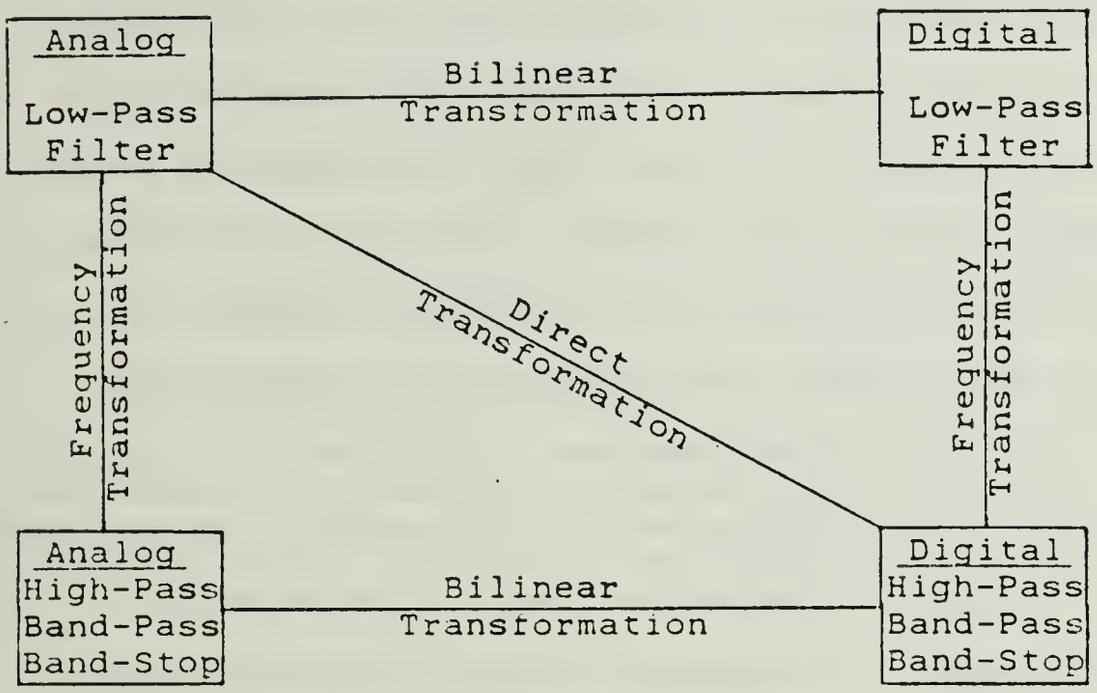


Figure 4.1 Analog and Digital Frequency Transformations

It should be reiterated that our goal in this section will be to develop a digital filter of Infinite Impulse Response (IIR) characteristics. This means that our filter will use the results of previous outputs to realize a later output. Although Finite Impulse Response (FIR) digital filters offer several qualitative advantages over IIR designs in the areas of phase linearity, stability, and an inherent protection against round-off error, they also require a larger number of delay elements to realize a design with a steep filter roll-off. This will be of concern to us when we realize an implementation in hardware with devices limited to a relative few number of filter transfer function poles and zeros.

A. ANALOG BAND-PASS TO DIGITAL BAND-PASS FILTER DESIGN

An analog-to-digital bilinear transformation makes it possible to apply a relation which transforms an analog band-pass filter into a desired band-pass digital design.

We will realize the 600 Hz digital bandpass filter by applying the bilinear transformation to the transfer function of the analog band-pass filter we developed in Chapter 3. Once implemented we will examine the performance of this filter in view of our goal of APU start-up identification. If necessary we will determine which refinements and modifications to our design may be necessary to realize our goal.

We recall from Chapter 3 that the transfer function of the analog elliptic band-pass filter was given by the product of the two second-order functions given by Eqs. 3.1 and 3.2. This product has been computed (as shown in Appendix C) and is found to be

$$\frac{V_2}{V_2} = \frac{\rho(s^4 + Fs^3 + Gs^2 + Hs + J)}{s^4 + Ms^3 + Ns^2 + Ps + Q}$$

where

$$\rho = 0.039811$$

$$F = 0$$

$$G = 29.587 \times 10^6$$

$$H = 0$$

$$J = 189.91 \times 10^{12}$$

$$M = 0.24351 \times 10^{3}$$

$$N = 27.642 \times 10^{6}$$

$$P = 3.3558 \times 10^{9}$$

$$Q = 189.91 \times 10^{12}$$

The poles of the analog band-pass filter transfer function were shown (graphically in Figure 3.5) to be:

$$-63.567 \pm 3.8421j \times 10^{3}$$

$$-58.188 \pm 3.5859j \times 10^{3}$$

The analog filter is therefore stable.

To transform the analog band-pass transfer function into a digital version we will use the Bilinear Transformation [Ref. 8: pp. 219-224] which is characterized by the following relation

$$s = \frac{2}{T}\frac{z - 1}{z + 1} \qquad 4.1$$

This transformation will map stable analog poles which are in the left-half of the s-plane into the interior of the unit circle in the z-plane. Thus stability is preserved in all cases.

If we then make the substitution $s = j\bar{\omega}$ and $z = e^{j\omega T}$ into Eq. 4.1 and simplify, we can establish the relationship between the frequencies in the analog and digital cases. (In this and the following discussion we shall denote frequencies in the analog case with an overbar ($\bar{\omega}$), and those in the digital case without one (ω).)

The resulting relation is

$$\bar{\omega} = \frac{2}{T} \tan \frac{\omega T}{2} \qquad 4.2$$

where T is the sample period given by $1/f_s$. In this case $f_s = (6.666 \times 10^6)/(4 \times 192)$, which we will show shortly. This results in $T = 1.15212 \times 10^{-4}$.

This relationship between analog and digital frequencies is shown in Figure 4.2 and reveals that the Bilinear Transform does not provide a linear mapping from one function to another. The frequency range from 0 to ∞ in the continuous case is warped into the frequency range from 0 to π/T in the digital case.

Therefore, if we have an analog filter with transfer function H(s), we may then perform the following substitution dictated by Eq. 4.1

$$H(z) = \bar{H}(s)\Big|_{s=(2/T)[(z-1)/(z+1)]} \qquad 4.3$$

Another way of expressing this same relation is

$$H(e^{j\omega T}) = \bar{H}(j\bar{\omega})\,|_{\bar{\omega} = (2/T)\tan \omega T/2}$$

Using this relation the characteristics of H(z) can be obtained graphically from those of $\bar{H}(s)$ as shown in Figure 4.2 [Ref. 5: p. 262].

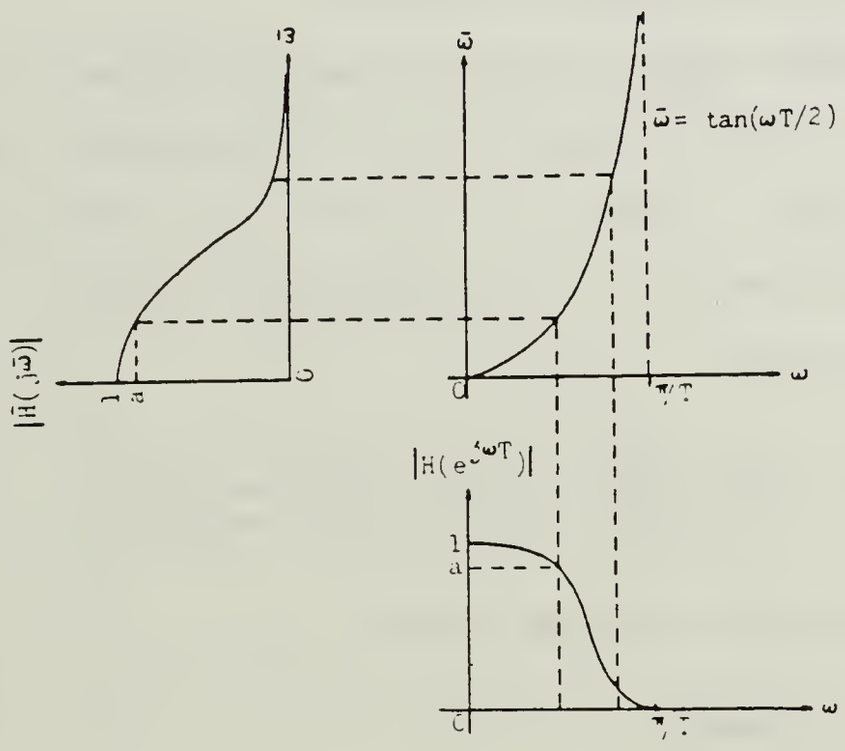


Figure 4.2 The Bilinear Transformation (showing analog and digital transfer functions and the non-linear warping of frequencies.)

We see from the figure that there is no aliasing problem associated with the transform because the frequency is limited to less than π/T (8680π) in the digital case. However, because of the frequency warping we have to make a proper transform of frequency according to Eq. 4.2

before application of the Bilinear Transform. Consequently, for a transformation of the analog band-pass filter derived as Eq. 3.3, we must substitute f_o = 590.825 Hz for f_o = 600 Hz before application of the Bilinear Transform to ensure a proper transformation to the digital domain. Once this is accomplished all we need do is apply the Bilinear Transform to the resulting "pre-warped" analog band-pass filter transfer function.

The FORTRAN based computer program previously introduced in Appendix C also provides for this development and implements Eqs. 4.2 and 4.3 to derive the following digital transfer function $H(z^{-1})$ for the desired digital band-pass filter

$$H(z^{-1}) = \frac{\rho(1 + Fz^{-1} + Gz^{-2} + Hz^{-3} + Jz^{-4})}{1 + Mz^{-1} + Nz^{-2} + Pz^{-3} + Qz^{-4}} \qquad 4.4$$

where the constants are as follows

ρ = 0.039516

F = -3.6279

G = 5.2861

H = -3.6279

J = 1.0000

M = -3.6251

N = 5.2586

P = -3.5768

Q = 0.97353

The poles of the transfer function given by Eq. 4.4 are

0.90781 ± 0.40813j (Magnitude = .99533)

0.90475 ± 0.40493j (Magnitude = .99123).

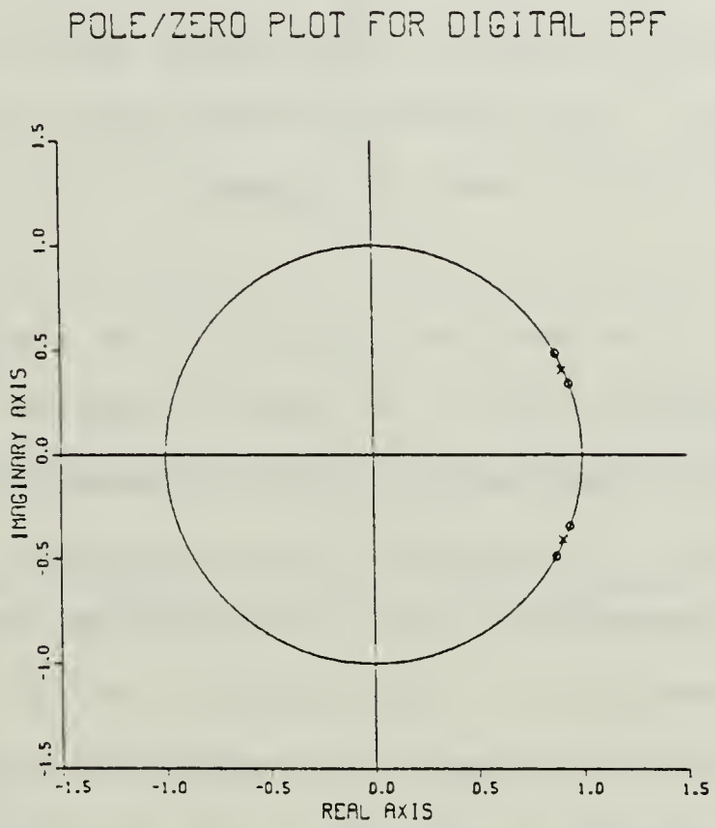


Figure 4.3 Pole/Zero Plot for the Digital Elliptic Band-Pass Filter (poles appear singular, but are in fact double and nearly coincident)

and thus we confirm the mapping of stable poles in the analog domain into stable digital poles located inside the

unit circle. This digital pole/zero plot is shown graphically in Figure 4.3 on the previous page.

B. DIGITAL BAND-PASS FILTER SIMULATION

Equation 4.4 represents the digital filter transfer function equivalent to the analog filter transfer function we presented in Chapter 3. In a manner completely analogous to that development we are now able to demonstrate a computer graphical simulation of the digital filter frequency and phase response and compare these to the previous results. The FORTRAN program used to present this graphical output is included in Appendix E under the title DBPFR.

In Figure 4.4 we see the digital filter frequency response and observe that it is nearly identical to the analog response in consonance with our design goal. The minor differences are remarkable and explicable. The center frequency of the digital filter is diminished to the pre-warped center frequency of approximately 591 Hz. Additionally, the two peaks of the amplitude response located at about 585 Hz and 595 Hz are not of equal magnitude. This is due to the difference of pole proximity to the unit circle. Although the poles appear coincidental in the graphical presentation in Figure 4.3, they are actually distinct; the pole nearer to the real axis is some

.004 units closer to the unit circle which accounts for the amplitude disparity between the two poles.

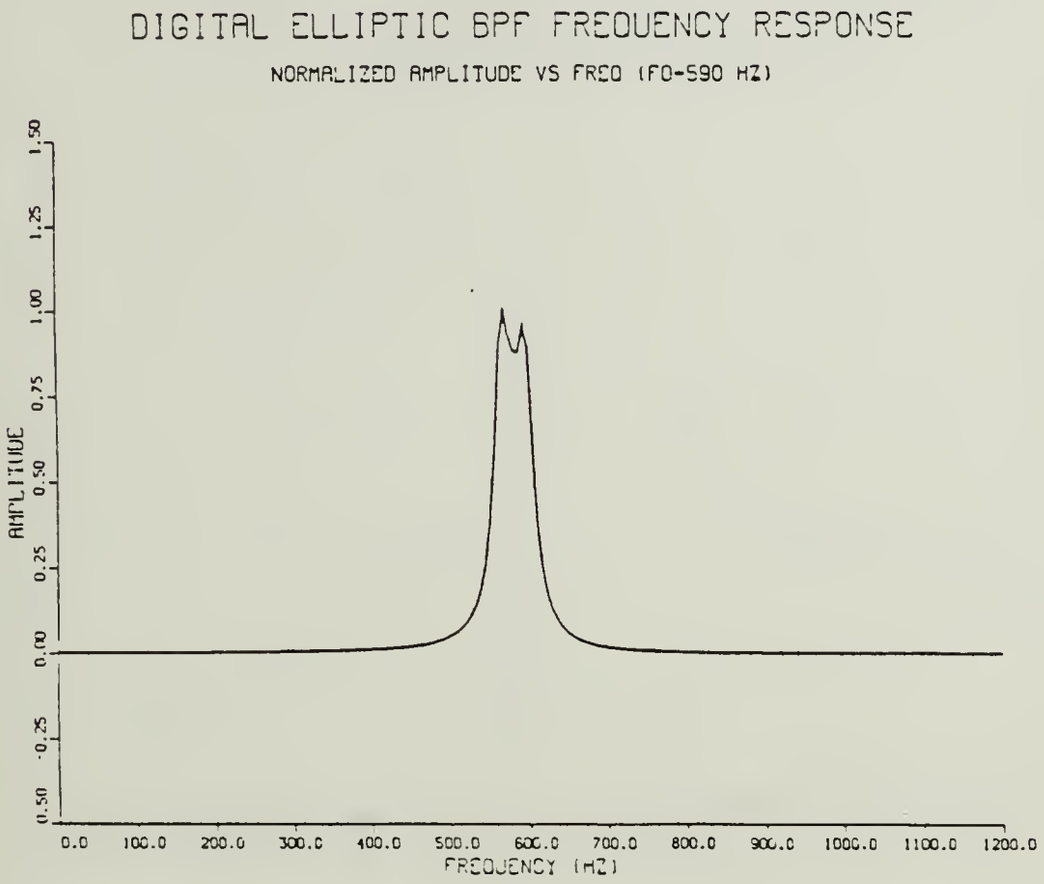


Figure 4.4 Digital Elliptic BPF Frequency Response (Computer Simulated Amplitude Response)

In Figure 4.5 we observe the digital filter frequency response as measured in dB. This curve appears somewhat different from its analog counterpart but the important feature is maintained. A steep filter rolloff is realized out of the passband and the response is diminished by about 30 dB at approximately 500 Hz and 700 Hz according to design specifications. Although the analog filter did not deviate much from this 30 dB down figure we see an added benefit of

the digital filter wherein the rolloff continues monotonically over our observed spectrum.

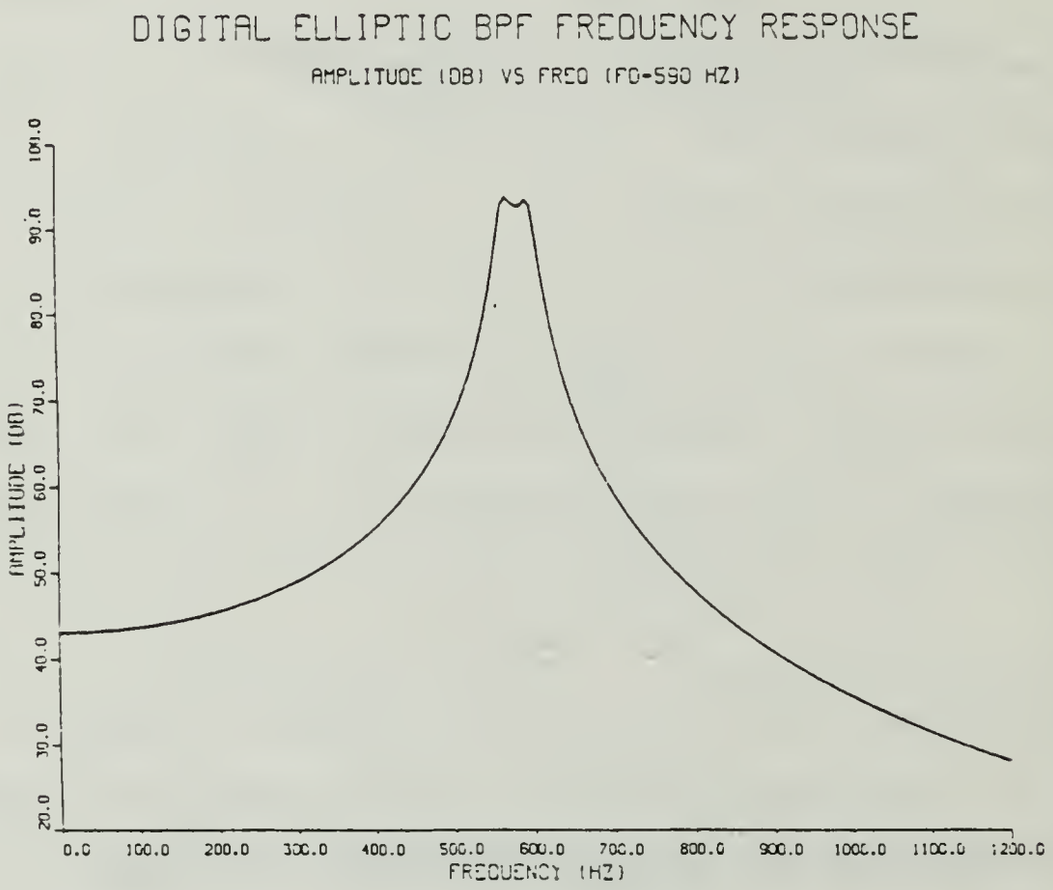


Figure 4.5 Digital Elliptic BPF Frequency Response (Computer Simulated Amplitude Response in dB)

Finally, in Figure 4.6 we observe the phase response of our digital filter. Once again this closely approximates the severe phase distortion we observed with the analog filter although the center frequency is again confirmed to be significantly less than the nominal 600 Hz we expected of the earlier filter design. To reiterate, this phase distortion is a hallmark of elliptic filters and the

Bilinear transformation, but our application is not phase dependent. Thus we may ignore this effect.

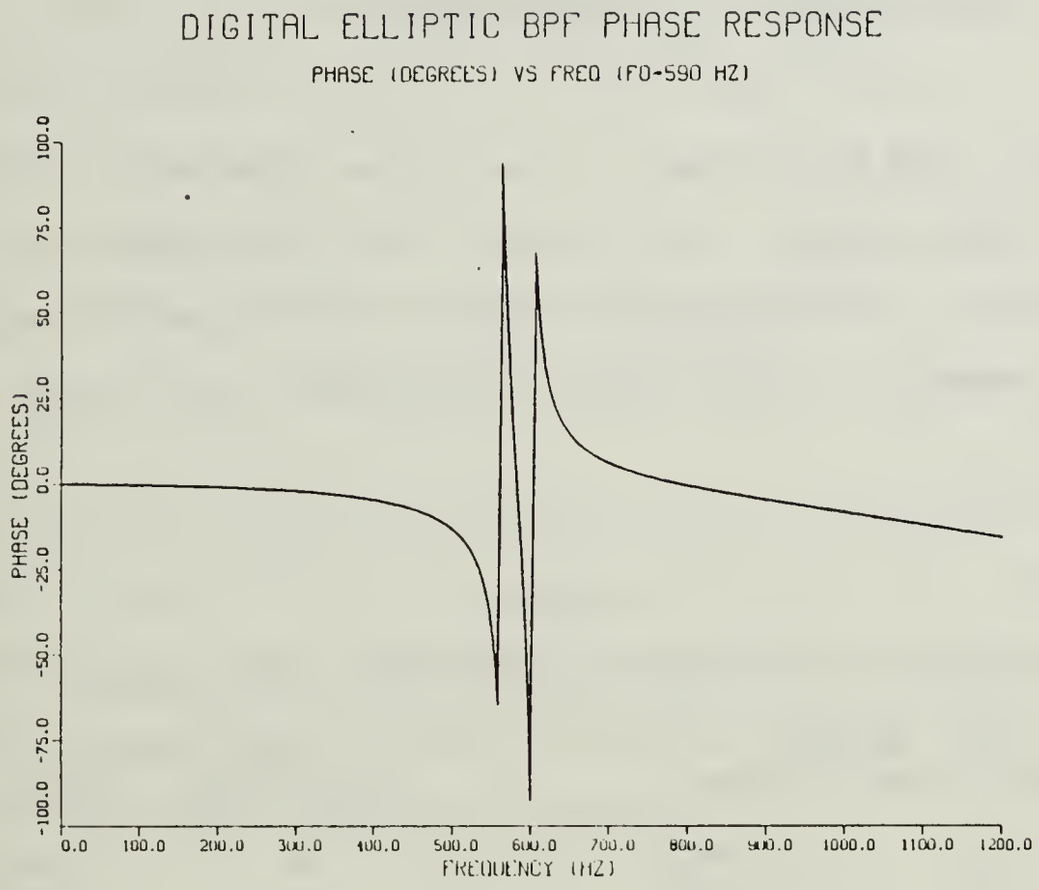


Figure 4.6 Digital Elliptic BPF Frequency Response (Computer Simulated Phase Response)

C. DIFFERENCE EQUATION REPRESENTATION

The transfer function for the fourth order Elliptic Filter was given previously in Equation 4.4. Section 4 of the program ABPDBP introduced earlier in Appendix C accomplishes a transformation of this quotient of fourth order polynomials and provides an equivalent cascaded

representation of two second order filter stage blocks, each of the form

$$H(z) = \frac{Y(z)}{X(z)} = \frac{a_0 + a_1 z^{-1} + a_2 z^{-2}}{b_0 + b_1 z^{-1} + b_2 z^{-2}}$$

where X(z) and Y(z) refer to the filter input and output, respectively. The values of the dual quadratic coefficients of the cascaded second order transfer function and as computed by the program in Appendix 4 are given in Table 4.1.

Table 4.1

ELLIPTIC BPF SECOND ORDER STAGE COEFFICIENTS

1st Stage Coefficient	Value	2nd Stage Coefficient	Value
a_0	1.0000	a_0	1.0000
a_1	-1.7503	a_1	-1.8780
a_2	1.0017	a_2	0.9981
b_0	1.0000	b_0	1.0000
b_1	-1.8163	b_1	-1.8092
b_2	0.9910	b_2	0.9822

Both second order z-domain filter stage transfer functions can be manipulated in a familiar way to realize the following z-domain difference equation.

$$b_0Y(z) = a_0X(z) + a_1X(z)z^{-1} + a_2X(z)z^{-2} - b_1Y(z)z^{-1} - b_2Y(z)z^{-2}$$

Applying the inverse z-transform to this z-domain difference equation yields the time domain digital difference equation .

$$b_0y(k) = a_0x(k) + a_1x(k-1) + a_2x(k-2) - b_1y(k-1) - b_2y(k-2) \qquad 4.5$$

The Signal Flow Graph corresponding to the difference equation given by Eq. 4.5 is shown in Figure 4.7. The difference equation representation is important because this is the basis for the hardware implementation of the digital filter which shall follow in Chapter 5.

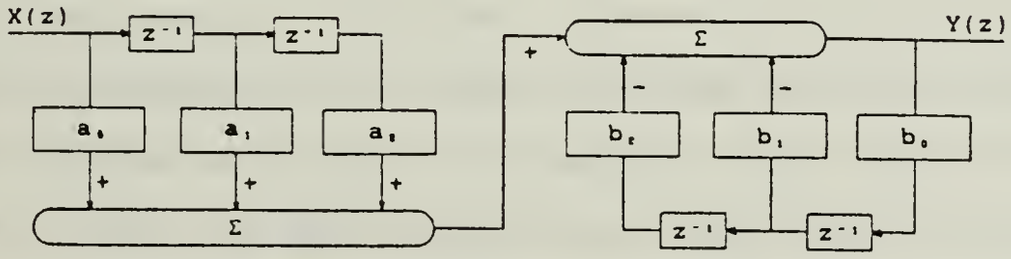


Figure 4.7 Second Order Stage Signal Flow Graph

D. DIFFERENCE EQUATION IMPLEMENTATION SIMULATION

We have just stated that the difference equation representation of the digital filter will become the basis for the INTEL 2920 Signal Processor implementation which is to follow. In order to show the adequacy of this method of implementation it is useful to demonstrate the impulse response and filter frequency response by computer simulation. In the following chapter sections we will demonstrate these simulations and show that they yield results in keeping with our design expectations.

1. Digital Filter Impulse Response

In Appendix F is presented the complementary FORTRAN programs S22I and S22IG. Both implement the impulse response simulation for the difference equation representation of the digital elliptic band-pass filter. In the case of S22I the output is digital and is shown to exemplify the filter response over a greater period of time than is usefully represented otherwise. In the case of S22IG the output is graphical and will be presented here.

In both these programs the impulse is equal to the greatest allowable input which guarantees an output of less than unity. This is done for reasons of filter stability as well as a limitation of the INTEL 2920 which will be discussed in the next chapter. By examining the impulse response we confirm the stability of the filter design by ensuring that the output decays to zero over time. In

addition we can observe the natural response of the system by establishing the frequency of the decaying sinusoid.

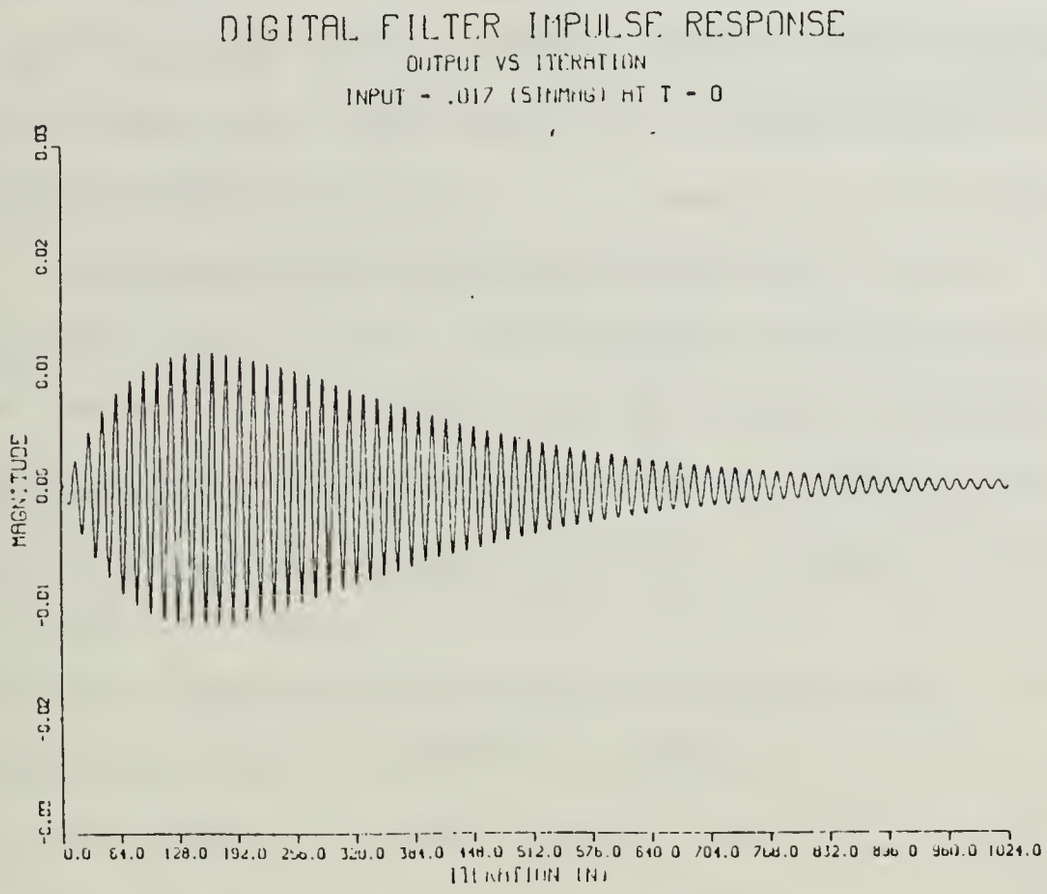


Figure 4.8 Digital Filter Impulse Response

In Figure 4.8 we observe the the response of the two stage difference equation filter for an input of 0.0172 applied at time T = 0. By counting the number of cycles which occur over a corresponding number of iterations, and realizing that the sample period is 0.11521 milliseconds we arrive at natural frequency which is very close to the expected 600 Hz. Additionally, it is apparently true that the response decays to zero with time, at least over the

approximately one-tenth of a second represented by the duration of the overall sample period in the figure. To confirm this suspicion we can carry out the impulse response for a substantially longer period of time, say over one full second, or approximately 8192 iterations. The results of this computation are shown in the output of S22I in the appendix. They confirm the occurence of the maximum amplitude of impulse response output at the xxxth iteration which is what we observe in the figure.

Having realized the digital filter impulse response output we can perform a discrete Fourier transform of this

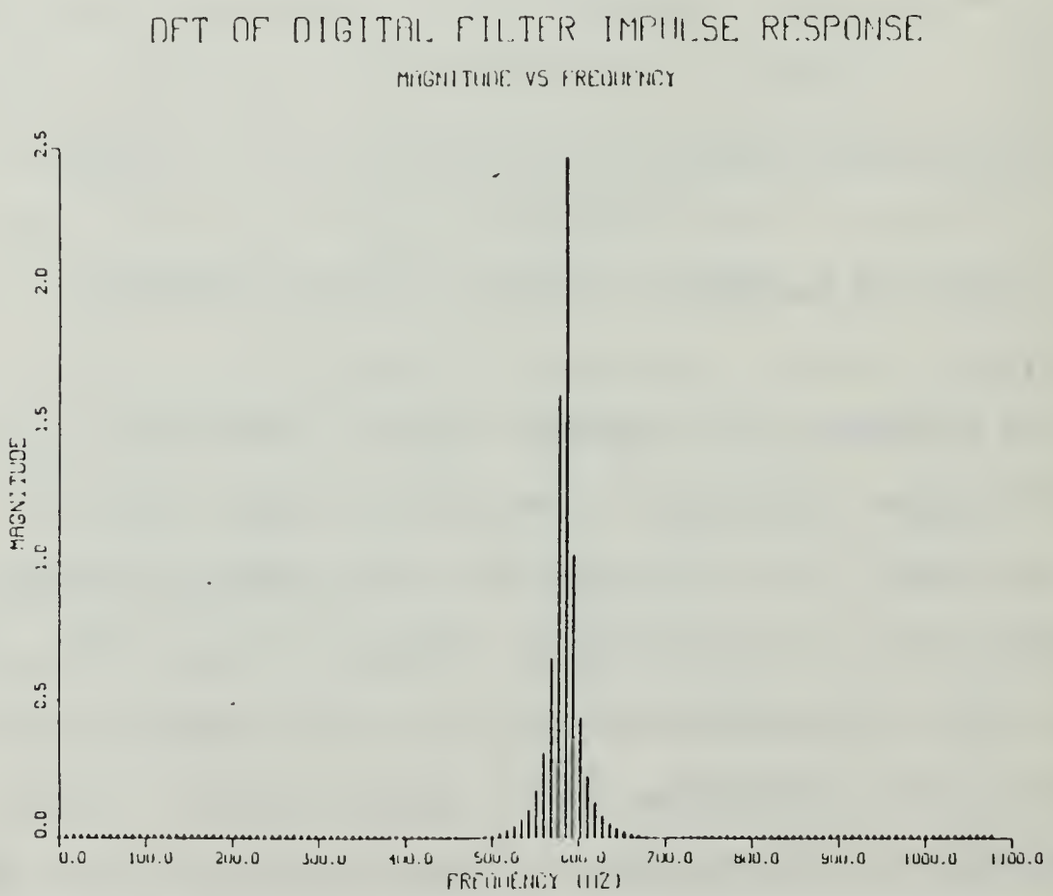


Figure 4.9 Discrete Fourier Transform of the Digital Filter Impulse Response

data to recreate the frequency response. This is shown in Figure 4.9. The figure confirms a narrow band-pass filter function with a center frequency at approximately 585 Hz. This is very close to the design center frequency of 591 Hz and is, in fact, within the error of a single bar in this pattern representation.

2. Digital Filter Frequency Response

Now that we have confirmed the stability of our filter design we can proceed to examine the frequency response of the filter over the range of interest. In particular we shall examine the filter frequency response over several frequencies in the range of 500 Hz to 700 Hz. The FORTRAN programs which allow this examination are S22F and S22FG which are included as Appendix G to this thesis. Due to the number of output figures they will be left in the appendix and we shall only give a summary of their content.

The digital filter frequency response was examined for the following frequencies: 500, 575, 590.825, 600, 625 and 700 Hz. The 590.825 simulation was chosen because this is the design center frequency (due to pre-warping) and we wish to confirm an output maximum amplitude at this frequency. From the figures in the appendix it is easy to see that the filter does in fact yield the response we desire. The maximum output amplitude does occur for the expected frequency, although the output at 575 Hz does not diminish appreciably from this value. However, we observe

output at 600 Hz is significantly less than at 590 Hz and even 575 Hz. This is indicative of both a steeper filter rolloff at frequencies greater than the center frequency and the effect of coefficient approximation which will be discussed more fully in the next chapter. The frequency response at both 500 Hz and 700 Hz is expectedly minimal but may not be usably low. If we find that the filter rolloff is not great enough and the response out of the passband is too great for our purposes then further design modifications may be undertaken. Accordingly, we could increase the order of our filter design. This would increase the number of filter stages in the analog implementation and therefore the complexity of that design. But, as we shall see, to a certain extent this additional filter complexity in the difference equation may be absorbed by the digital implementation we shall pursue without any increase in the hardware. These considerations will have to be examined more completely in the final analysis of the filter design effectiveness.

E. ANTI-ALIASING FILTER

When implementing a digital filter it is necessary to employ an analog input anti-aliasing filter to limit the band of input frequencies to less than half of the Nyquist sampling rate. This corresponds to the need to implement a low-pass filter at the input to the digital band-pass

filter. We shall see in the discussion which follows that the sampling frequency will be 8680 Hz. Thus our band of input frequencies is limited to less than one-half of this value, or 4340 Hz. Because our frequency of interest is 600 Hz we have considerable freedom in choosing the cut-off frequency of our anti-aliasing filter.

One option available to us is to design a low-pass filter with a rolloff which meets our needs. However, there are such filters commercially available which implement a compatible response which minimizes the effort required of the designer. One such filter is the INTEL 2912A which has been specifically included in the hardware kit we shall use to implement the digital filter we have just developed. This hardware implementation is the subject of Chapter 5.

V. THE INTEL 2920 ANALOG SIGNAL PROCESSOR

A. OVERVIEW

The INTEL 2920 Analog Signal Processor is actually a digital processor which is implemented to perform analog signal processing functions. Introduced in 1979, the 2920 system is centered about the 2920 single-chip microcomputer which is specially designed to process real-time analog signals. This single chip includes within its 28-pin DIP configuration sufficient hardware to provide 192 program memory locations, scratchpad memory, digital to analog (D/A) circuitry for up to four separate sampled inputs, analog to digital (A/D) capabilities for eight individual outputs, a digital pipeline processor capable of up to twenty-five bit accuracy, and input/output (I/O) control circuitry [Ref. 7: p. 3-1 through 3-2] . The 2920 is capable of implementing a wide variety of functions which rely upon sampled digital data techniques. We will use the 2920 to implement our matched filter design which will detect the APU start.

At the heart of the 2920's significant power is its on-board erasable programmable read-only memory (EPROM) which allows the user the convenience of customizing the 2920 for each intended application. Because the 2920 is a pipeline processor all program steps are performed sequentially without any conditions which may impact upon execution time.

Several logical conditions are allowed which affect program manipulation of data, but none will cause the processor to execute a program step out of sequence. In fact the only effective jump is performed at the last instruction which provides for a return to the beginning of the program loop. In this way the programmer may provide for an exact digital sample interval based upon program loop execution time. The shorter the program implementation loop the greater is the processor capacity to provide a faster sampling frequency.

The necessity of providing for an accurate sampling interval arises out of an understanding of the characteristics of the sampled analog signal being processed. Without an accurate clock interval, provided for in the 2920 by the program execution loop time, significant noise can be introduced in the system. Even small variations in the sampling interval can render the analysis useless through the introduction of intolerable measurement noise.

Each 2920 program instruction requires four clock cycles to execute. Given our nominal 6.666 MHz clock (the maximum allowed) the 2920 can therefore realize a maximum sampling frequency of 8.680 kHz over a 192 instruction program loop. This allows for a device bandwidth of greater than 4 kHz. Shorter programs naturally allow for a greater sampling frequency and thus higher device bandwidth.

In addition to the precision and speed of computation which the 2920 offers, it also allows for sequential processing of up to four separate input signals and eight analog output signals in a single program pass. This is of course dependent upon program complexity--regardless of

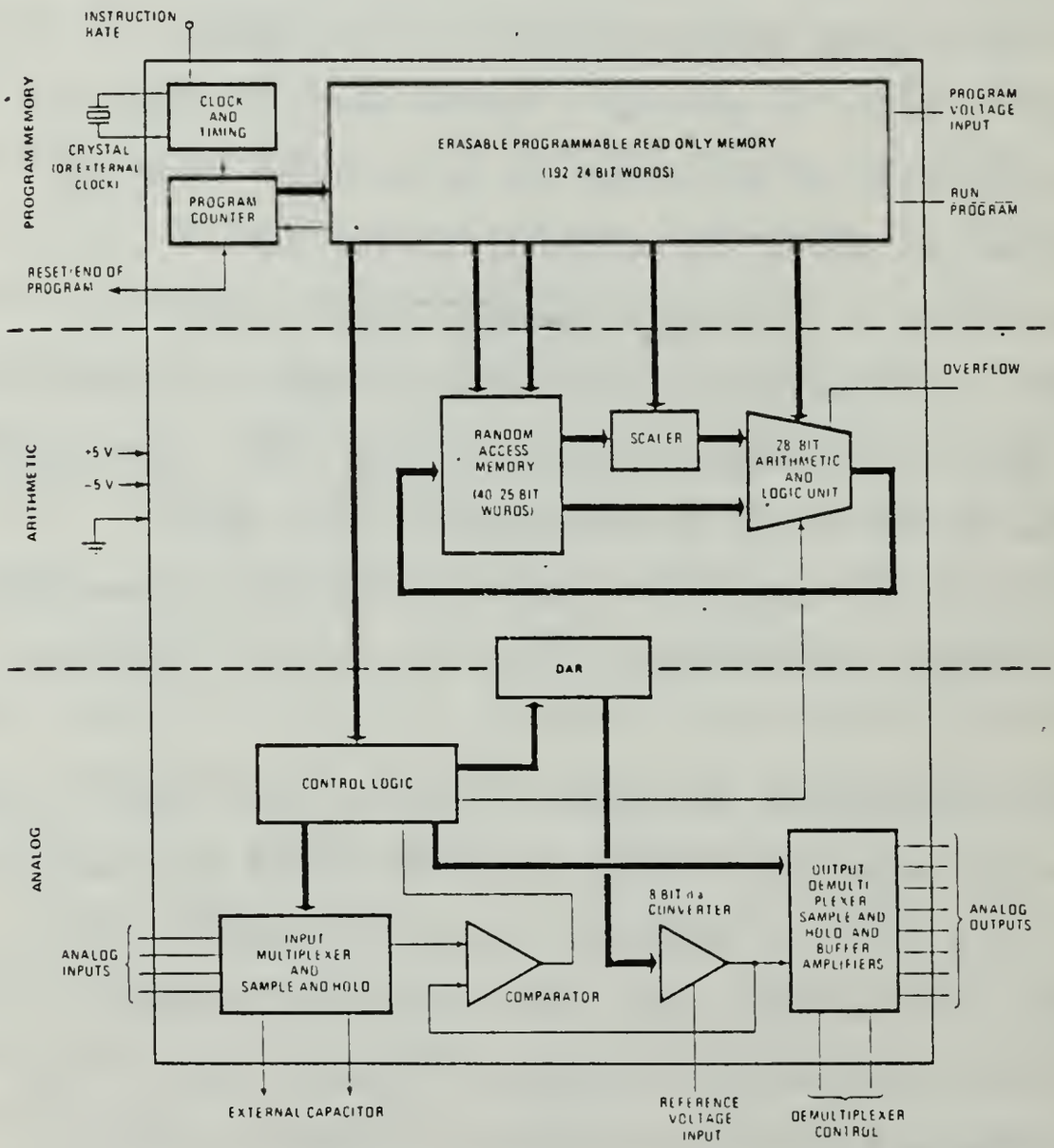


Figure 5.1 2920 Functional Block Diagram

function, any applications program cannot make use of more than 192 instructions to process whatever number of input and output signals are being manipulated. But despite this restriction the power of the 2920 is evident. In our application we will only make use of a single input/output channel.

B. 2920 FUNCTIONAL DESCRIPTION

Figure 5.1 on the previous page details the block configuration of the 2920 [Ref. 9]. It is divided into the three major subsections described as follows.

The 192 x 24-bit Program Memory Section is a storage area implemented with EPROM. This section includes the instruction clock and timing circuits and program counter which control the operation of the entire device, including the other two sections.

The Arithmetic Section includes a 40 word by 25-bit scratchpad RAM and an arithmetic and logic unit (ALU). Both the RAM and the ALU are two port access devices. In the case of the ALU one of the ports is passed through a barrel shifter scaler. The function of the arithmetic section of the 2920 is to execute the commands dictated by the program memory.

The Analog Section performs A/D and D/A functions upon command from the program memory. In order to implement

these functions the analog section includes the following subsections:

-- a four input multiplexer

-- an input sample-and-hold circuit

-- a D/A converter

-- a comparator

-- an output multiplexer with eight output sample-and-hold and buffer amplifiers.

-- a special digital-to-analog (DAR) register which acts as an interface between the digital and analog sections.

C. THE SDK-2920 DEVELOPMENT SYSTEM

The SDK-2920 Development System is an integral component in the development of any applications package which uses at its core the INTEL 2920 Analog Signal Processor [Ref. 10]. Within the scope of the system are many development capabilities including

-- Breadboarding: The breadboard is used to develop circuits for evaluation or prototype applications.

-- Assembling and Editing: This feature is comprised of an assembler, disassembler, hexadecimal display, symbolic 2920 instruction display, and single keystroke entry of many 2920 instruction fields.

-- 2920 EPROM Programming: The development board includes hardware and control elements necessary to program the 2920.

-- Communications: The development also interfaces with Intel Developments Systems (such as the Intellec Series) to pass object and source code listings of 2920 programs.

Applications software was developed using an Intellec Microcomputer Development System running a 2920 Assembly Program and Software Simulator. Transfer of 2920 software between the development system and the SDK-2920 is easily accomplished.

The SDK-2920 is physically divided into a development side and an applications side. The development side can be used to load, test and modify EPROM resident programs under 8085A microprocessor control. System control is accomplished with the use of a keypad monitor. The composition and hierarchy of the monitor command structure is shown in Figure 5.2 on the following page.

The applications side includes a prototype area for circuit construction and testing. It functions independently of the development side. After program development has taken place two methods may be used to accomplish program verification. The first method uses the Intel SM2920 Simulator Software to simulate the execution of programs written for the 2920. This simulator allows the use of symbolic references for changing and displaying all 2920 registers, flags and user-defined locations in program and memory storage. A trace feature also allows monitoring of selected parameters as they are changed under program control.

The second method of 2920 program verification is done by monitoring circuit performance on the applications side

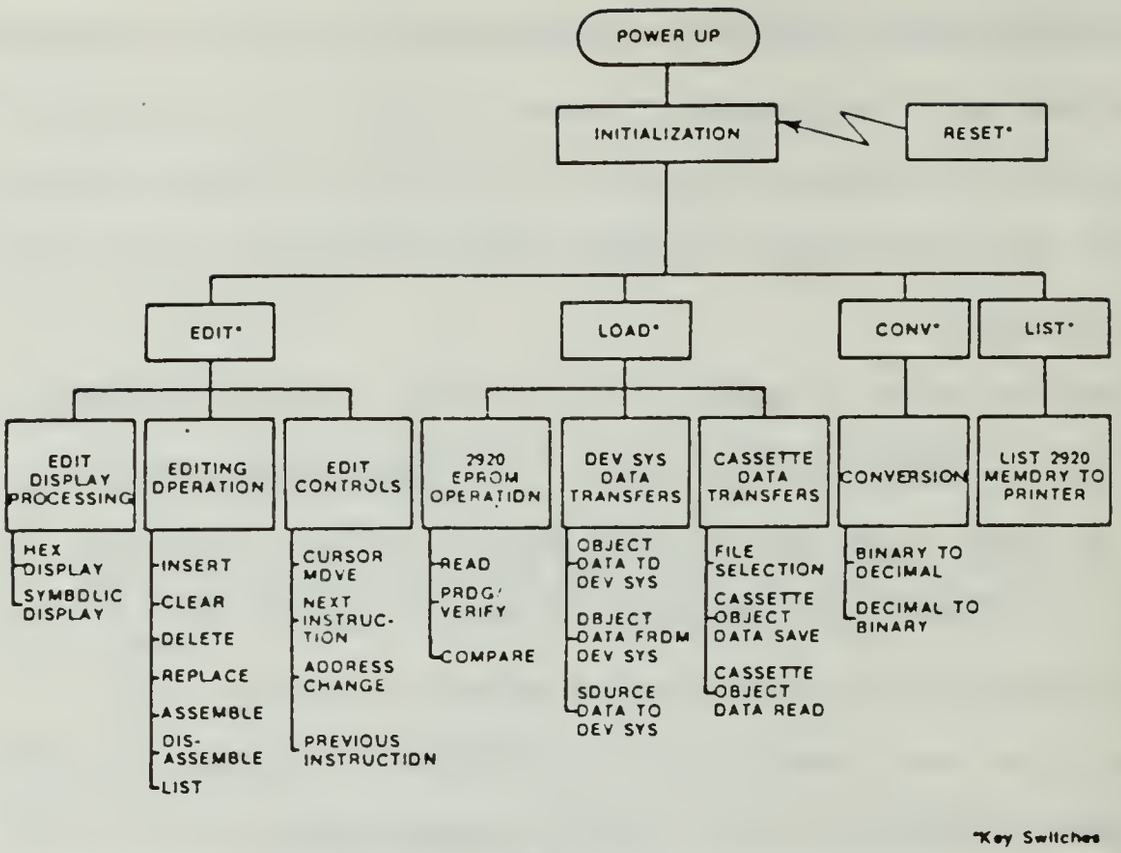


Figure 5.2 SDK-2920 Monitor Command Structure

of the SDK-2920 by examination of inputs and outputs. The applications board is shown in Figure 5.3 on the page following. Provisions are made on the board for assembly of either internal or external clocks, four input and four output channels with associated waveshaping circuitry, reference voltage development, and two user breadboard areas for specific applications development. Furthermore, TTL compatible output signals can be delivered to the output vice analog outputs if desired. We shall make use of this feature to send a signal detection pulse when the APU ignition is detected.

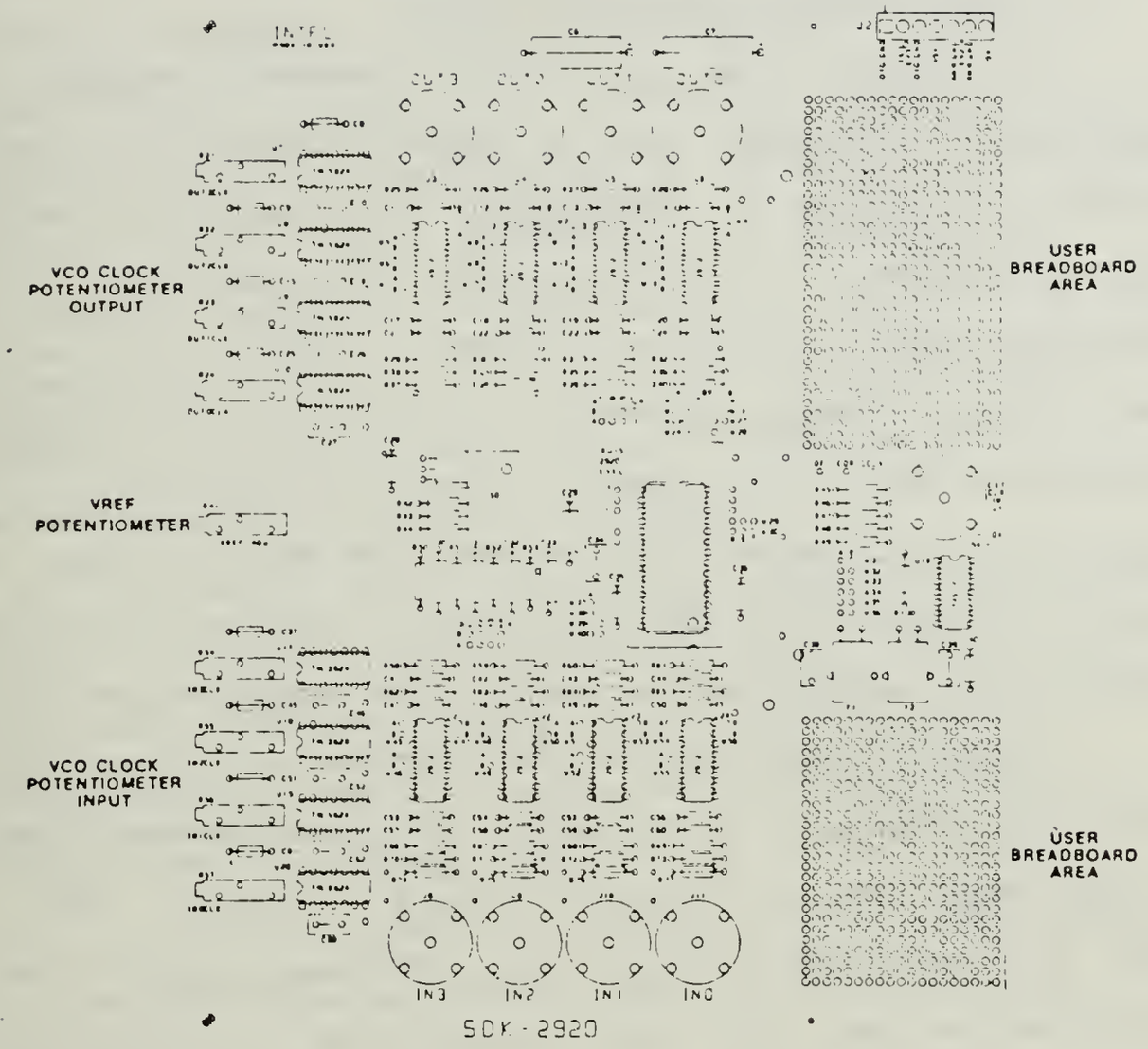


Figure 5.3 SDK-2920 Applications Board

D. A 2920 DIGITAL FILTER IMPLEMENTATION

Here we shall describe the particular 2920 software and hardware components which comprise the digital filter.

1. 2920 Assembly Language Program

In Appendix H we find the 2920 assembly language program which implements the two stage difference equation developed in Chapter 4. Recognizing the characteristics of the 2920 processor, it is instructive to review the programming devices which are brought to bear to realize this filter. We will proceed in the order in which these devices are used in the program. Appendix H should be consulted as reference for the discussion which follows. A detailed discussion of the 2920 Assembly Language should be consulted for particulars concerning the language [Ref. 11].

After initializing the DAR register we accept the input analog sample from the sensor microphone/preamplifier ensuring that the level does not exceed 1.0 volts. This limit is established by the voltage reference circuitry at pin #8 of the 2920. The input analog value is stored in the Sample/Hold register.

We then begin a sequence of steps, according to 2920 protocol, which accomplish the analog to digital transformation of the input value in the Sample/Hold register. This procedure is completed at program step 43 and the resulting digital value is then found in the DAR register.

Although we have ensured an input less than one volt this is not sufficient to guarantee that the post-processing value will not exceed the internal arithmetic limit. Internal arithmetic is limited to a range of values which cannot exceed -1.00000000 to +0.99999999. These 8 decimal place accuracies are established by the internal 25 binary bits (1 sign bit and 24 magnitude bits) available for arithmetic computations within the 2920. Actually the range of multiplicative inputs is only good to within 4 decimal places due to the scaling problem. But this will be seen to be more than adequate for our purposes.

To ensure that the processed values do not exceed one volt we scale down the digitally sampled input value by 64 by way of program step 44. The difference equation manipulations of the input value are then accomplished in program steps 47 through 130.

Digital arithmetic is performed in the 2920 by means of binary shifting and adding which is predicated on a transformation of coefficients to a nearest binary equivalent. The FORTRAN program and its output which performs this transformation is labelled CTRANS2 and is shown in Appendix I. Although a binary transformation does involve some approximation error, we see in the appendix that the worst case approximation of coefficients is still within .02 percent of the actual value. This is a relatively insignificant error.

After the difference equation implementation in each program pass we are left with a binary value in the DAR register which corresponds to the program output for that pass. We have the option of providing a certain amount of linear output gain by an appropriate shift of the output binary value now in the DAR. In program step 132 we provide a gain of four by a left shift of two binary positions. We output this value to channel 0 in steps 139 through 142.

The final program manipulation occurs in program steps 143 to 150. Here we perform a serial register shift of present program pass values in preparation for the next program pass. Program step 191 is the final executable statement which returns us to step 0 for the next pass. The entirety of the 2920 operation consists of an endless loop of these instructions.

2. 2920 Hardware Implementation

The 2920 contains an EPROM which is loaded with the hexadecimal code which is equivalent to the assembly language program just described. However, there are several other component devices which are integral to the operation of the 2920. The relation of these devices to the 2920 will now be described. A graphical schematic of these components appears in Figure 5.4.

At the input side of the 2920 an anti-aliasing filter is realized by using a 2912A which actually contains two filters which are cascaded together. This configuration

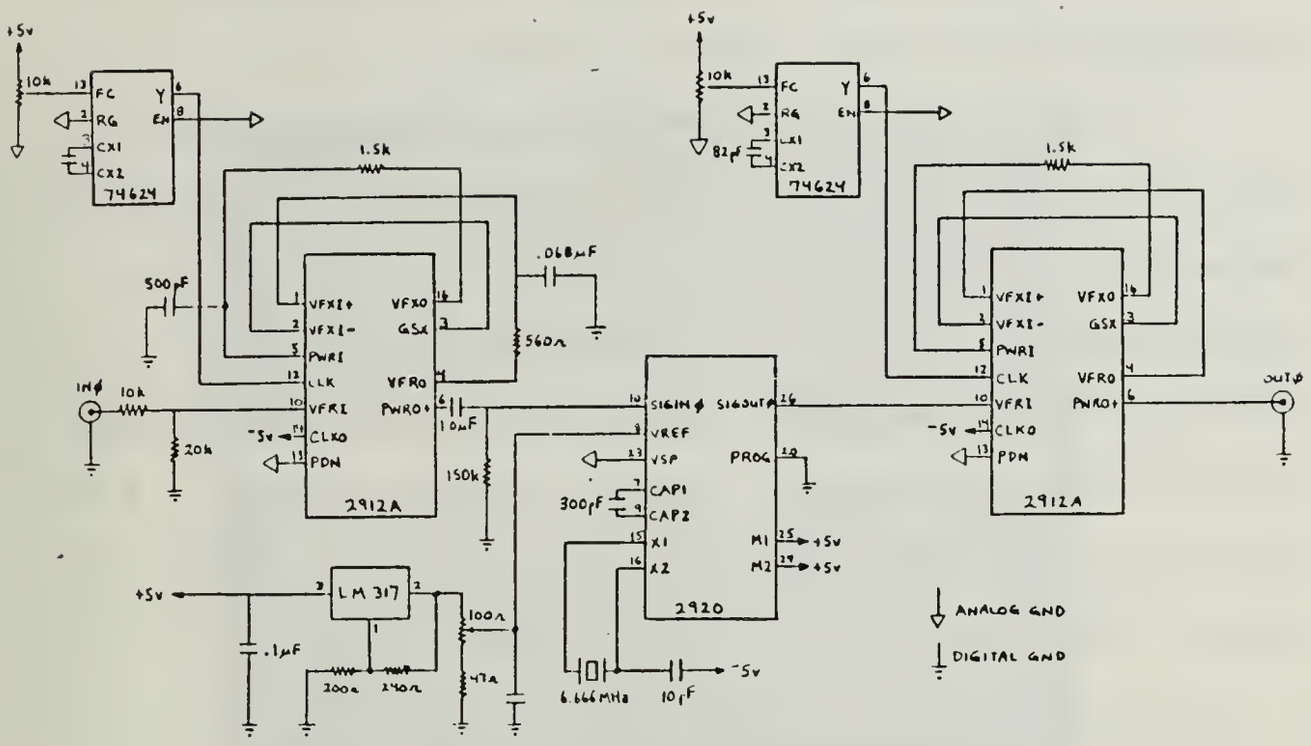


Figure 5.4 2920 Digital Filter Schematic

provides 54 dB of input dynamic range and a nearly flat response for frequencies less than 3 kHz. There a steep roll-off commences and at about 4 kHz the cascaded filter combination provides over 30 dB of attenuation. This supports the Nyquist frequency limit which is 4.34 kHz in this application.

The 2912A is a pulse code modulated filter which requires a clocking input to realize its filter function. This is provided by the 74624 at its input.

At the output of the 2920 another 2912A is employed in identical configuration and now provides a reconstruction filter for our application. This filter smooths the output

of the 2920 to provide an analog signal for follow-on logic discrimination as shown earlier in Chapter 3.

A 2920 option which is not demonstrated here yet will be employed in final filter configuration is to obviate the need for external signal conditioning by allowing program discrimination of the output value and thus providing a processed TTL signal output. This eliminates the need for the external circuitry shown in Figure 3.8 and therefore represents one significant advantage of the 2920 digital design over the analog implementation.

E. 2920 DIGITAL FILTER IMPLEMENTATION RESULTS

We will now proceed to demonstrate the results of the 2920 digital filter implementation in much the same manner as the presentation which accompanied the analog filter design in Chapter 3. We begin with a photograph of the digital filter frequency response to a ramped sinusoidal input. This is shown in Figure 5.5. The same method was used to generate the sweep oscillation although the range of sweep is not identical to that employed in generating Figure 3.9. The result is that we cannot guarantee the narrow bandwidth of this digital filter relative to its analog counterpart by this means alone. The intent is, as before, only to demonstrate that a narrow band-pass filter response is generated.

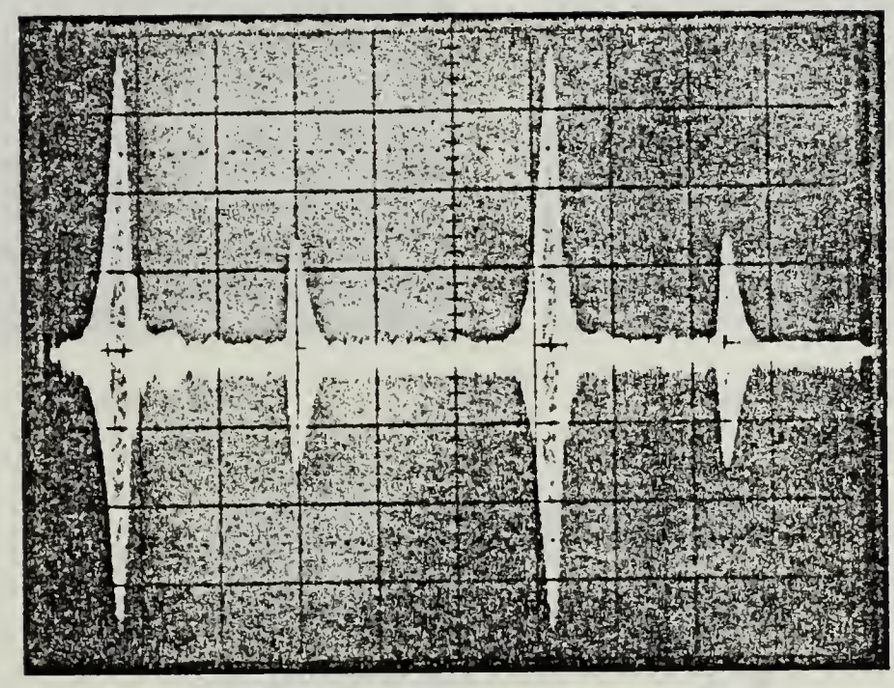
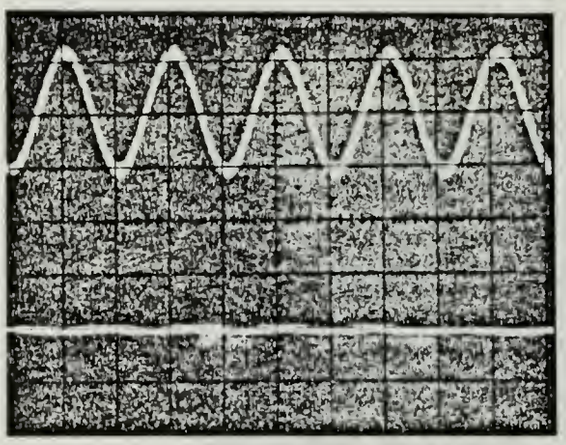
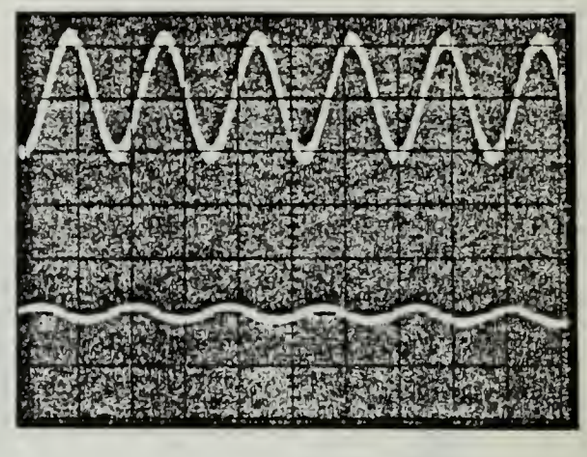


Figure 5.5 Digital Elliptic BPF Frequency Response
(Photograph of Actual Filter Amplitude Response
to a Ramped Sinusoidal Input)

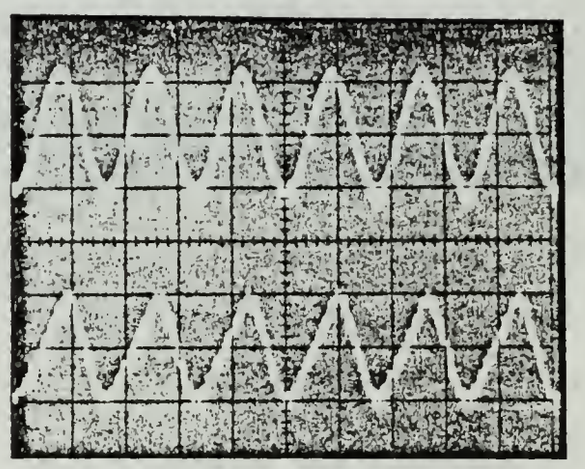
To confirm the operation at the desired band-pass center frequency we next apply discrete sinusoidal inputs to the digital filter at various frequencies arrayed about 600 Hz. The result is the digital analog to Figure 3.10 which is shown here as Figure 5.6. The scale is maintained as in Figure 3.10. The input frequencies are at about double the amplitude of the analog filter to ensure proper operation. This implies that despite the relative immunity of the digital filter to input amplitude variations we must nontheless provide an input above approximately 100 millivolts peak-to-peak. However, once above this threshold



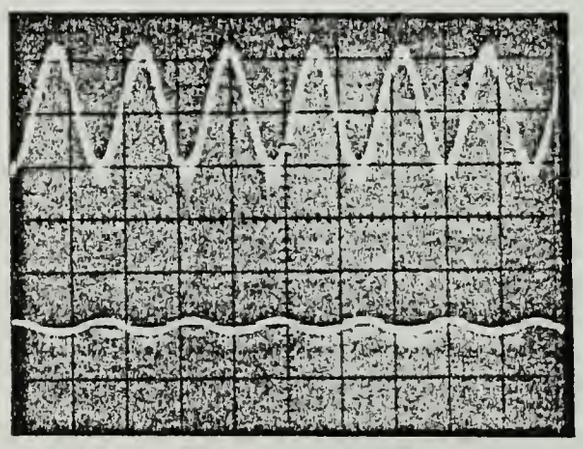
a) 500 Hz



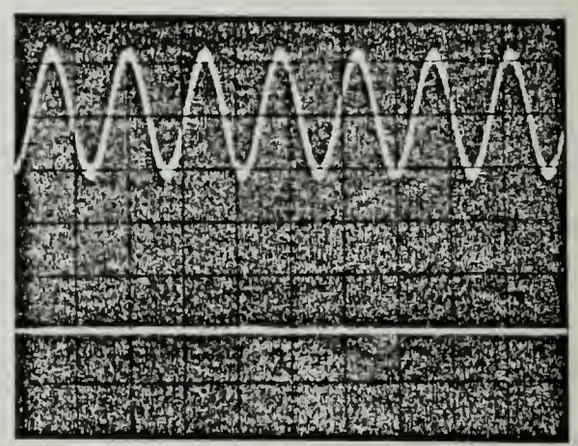
b) 575 Hz



c) 600 Hz



a) 625 Hz



b) 700 Hz

Figure 5.6 Digital Elliptic BPF Frequency Response
Upper trace (Input): 50 mV/div scale
Lower trace (Output): 1.0 V/div scale

value the digital filter provided a relatively undistorted and largely constant amplitude output up to an input amplitude of over 5 volts peak-to-peak (and this despite the 1 volt reference level of the input). Figure 5.6 thus confirms the center frequency maximum at a value near 600 Hz and a steep roll-off on either side of this value.

VI. ALTERNATIVE FILTER CONCEPTS

The preceding development was based upon techniques used in implementing an Infinite Impulse Response (IIR) digital filter. Simply stated, an IIR filter realizes its output based upon the values of all present and previous inputs and outputs. In other words, feedback is employed in an IIR design.

In the general case, an IIR filter will have M finite zeroes and N finite poles. The zeroes of H(z) can be anywhere in the z-plane but the poles must lie within the unit circle to guarantee stability. In the case we have developed, a digital filter realization derived from an analog design, the order of M must be less than or equal to N. This describes an Nth order digital filter.

The hardware implementation of an IIR design usually involves the cascading of elemental single pole filters with double complex pole filters. These elements are derived from the original transfer function using a partial fraction expansion separation scheme.

There are other methods for realizing the filter we desire other than the a priori scheme we have developed so far. These generally use the input signal itself as a basis for the filter transfer function coefficients and involve an

adaptive evaluation of the proper coefficients which yield the desired filter response.

A. A WIENER FILTER DESIGN--THE ADAPTIVE LINEAR COMBINER

The Adaptive Linear Combiner (ALC), shown in Figure 6.1, forms the basis for the Adaptive Filter design we shall now discuss [Ref. 12]. An input analog signal may be digitally sampled in accordance with the Nyquist criterion and we may then apply N sequential elements of that sample block to

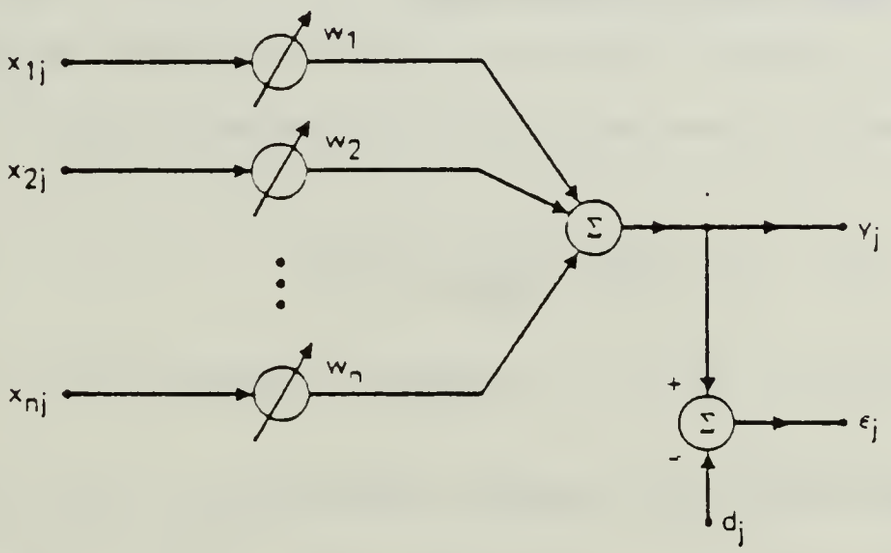


Figure 6.1 The Adaptive Linear Combiner

the ALC inputs. These inputs can be easily derived from a tapped delay line which cascades sample values along in sequential storage for processing. This scheme lends itself well to implementation in a processor such as the 2920 which is designed to accept sequential values by way of its component A/D converter and RAM storage.

The set of measurements x_{nj} is multiplied by a corresponding weighting term W_i, and the results then summed to yield the output y_j. This output is then compared to a desired signal value for that instant and the difference between them constitutes an error signal ϵ_j. The objective of the ALC is to determine W_i so as to minimize ϵ_j for each set of sampled inputs and thus realize the weighted sum of input signals that best matches the desired response.

1. Theoretical Foundations

At the nth instant of time the output of the Non-Recursive Wiener ALC, y(n), is given by [Ref. 13]:

$$y(n) = \sum_{j=0}^{N} x(n-j)W_j$$

$$= W_0x(n) + W_1x(n-1) + \ldots + W_Nx(n-N)$$

which may be written in matrix form as

$$= \underline{W}^T\underline{X}$$

or equivalently

$$= \underline{X}^T\underline{W}$$

where T represents the matrix transpose operator, the set of N+1 weights is denoted by

$$\underline{W}^T = [W_0\ W_1\ \ldots\ W_N]$$

and the set of present and N previous inputs is given by

$$\underline{X}^T = [x(n)\ x(n-1)\ \ldots\ x(n-N)]$$

The error signal e(n) for time n is given by

$$e(n) = d(n) - y(n)$$

$$= d(n) - \underline{W}^T\underline{X}$$

and the square of the error (using the latter matrix notation) by

$$e^2(n) = e(n) \cdot e^T(n)$$

$$= [d(n) - \underline{W}^T\underline{X}][d(n) - \underline{X}^T\underline{W}]$$

$$= d^2(n) - 2d(n)\underline{X}^T\underline{W} + \underline{W}^T\underline{X}\underline{X}^T\underline{W}$$

The mean square error, obtained by taking the expected value of this last equation, is given by [Ref. 13]

$$E[e^2(n)] = E[d^2(n)] - 2E[d(n)\underline{X}^T]\underline{W} + \underline{W}^T E[\underline{X}\underline{X}^T]\underline{W}$$

Defining the vector Φ_{xd} as the cross-correlation between d(n) and $\underline{X}$ then yields

$$\Phi_{xd} \equiv E[d(n)\underline{X}]$$

$$= E[d(n)x(n), d(n)x(n-1), \ldots, d(n)x(n-N)]^T$$

The input auto-correlation matrix Φ_{xx} is defined as

$$\Phi_{xx} = E[\underline{X}\underline{X}^T]$$

which may be written in expanded notation as

$$= \begin{bmatrix} x(n) \\ x(n-1) \\ x(n-2) \\ \vdots \\ x(n-N) \end{bmatrix} [x(n)\ x(n-1)\ \ldots\ x(n-N)]$$

Now if we carry out the indicated vector multiplication we arrive at the following result [Ref. 13]

$$= \begin{bmatrix} x(n)x(n) & x(n)x(n-1) & \ldots & \\ x(n-1)x(n) & x(n-1)x(n-1) & \ldots & \\ \vdots & \vdots & & \\ & & & x(n-1)x(n-1) \end{bmatrix}$$

And thus we arrive at the following form of the input correlation matrix

$$= \begin{bmatrix} \varphi_{xx}(0) & \varphi_{xx}(-1) & \cdots & \varphi_{xx}(-N) \\ \varphi_{xx}(1) & \varphi_{xx}(0) & \cdots & \varphi_{xx}(1-N) \\ \vdots & \vdots & & \vdots \\ \varphi_{xx}(N-1) & \varphi_{xx}(-N) & \cdots & \varphi_{xx}(-1) \\ \varphi_{xx}(N) & \varphi_{xx}(N-1) & \cdots & \varphi_{xx}(0) \end{bmatrix}$$

In order to find the optimal weight vector, $\underline{W}^*$, we can differentiate the mean square error function with respect to the weight vector $\underline{W}$ to yield

$$\frac{d(e^2(n))}{d\underline{W}} = -2[\Phi_{xd} - \Phi_{xx}\underline{W}]$$

The optimal weight vector , $\underline{W}^*$, generally called the Wiener weight vector, is obtained by setting the quantity in brackets equal to zero. This results in

$$\underline{W}^* = \Phi_{xx}^{-1}\Phi_{xd}$$

The objective of processes involving the ALC is to find a solution to this equation. In fact we may employ an adaptive algorithm which uses the error signal, $\epsilon(n)$, (generated for each instance of filter inputs), as the basis

for modifying the filter weights until a minimum error is attained for a particular input block. This describes the Adaptive Transversal Filter shown in Figure 6.2 [Ref. 12].

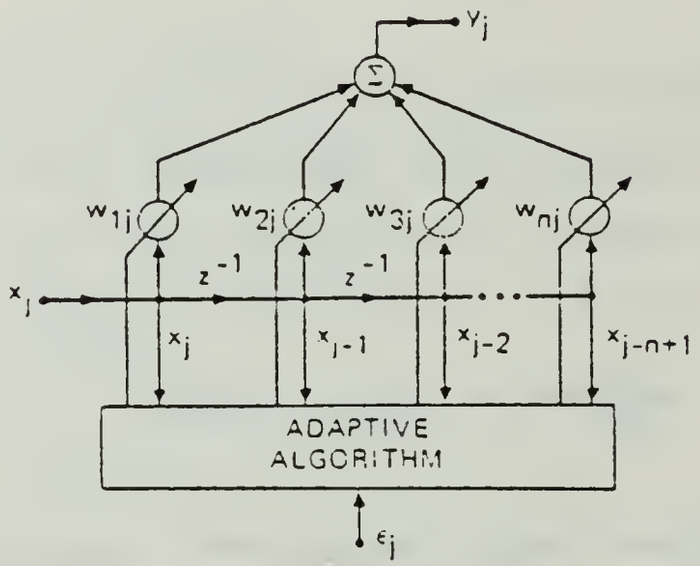


Figure 6.2 The Adaptive Transversal Filter

The Adaptive Transversal Filter (ATF) is a Finite Impulse Response (FIR) filter owing to the lack of direct feedback from output to input. If we employ a tapped delay line at the input to the ALC which comprises the ATF the form of the input vector becomes a finite number of delayed elements of the input signal. It is therefore easy to see that the impulse response of the ATF is just the sequence corresponding to the elements of the weight vector, $\underline{W}$. Such a filter can have any impulse response of length less than or equal to its own length. Allowing for an ideal unlimited length we could realize any impulse response at

all, and thus any frequency response. Practically, however, we are limited by filter complexity, error due to misadjustment, and an adaptive time constant which corresponds to filter length.

Thus we have a means of generating the desired filter response by applying the very signal we wish to detect. If we apply a digital series of samples taken from an analog reference signal we can realize the filter weights which will provide our desired signal output stream at a later time.

Thus the idea is to sample the analog recording of the APU noise in the cargo bay prior to launch and to apply that input series of data elements to an ATF to realize the filter weights. We may then build a 2920 circuit which uses these weights as filter coefficients to provide our filter response.

2. A Software Simulation

As an example of this methodology we will now present an elementary simulation which was performed for an input which consisted of an equal amplitude application of the three fundamental frequencies of interest: 600 Hz, 1200 Hz and 1800 Hz. We chose to simulate an Adaptive Transversal Filter of fourth order which therefore consists of four weights.

One example of a software implementation which is designed to arrive at the four desired filter weights is

shown in Appendix J as FORTRAN program FIR4. In this program we begin with trial weights and a range of upper and lower bounds. By repeated application of a library coefficient optimization routine (BOXPLX--also included in the appendix) we arrive at a set of four optimal weights within the bounds specified.

The result of this simulation is revealed through application of the FORTRAN program FIR4SIM which is included as Appendix K to this thesis. This result is shown in Figure 6.3. The input analog signal (indicated by the solid line) is a portion of the combined signal corresponding to

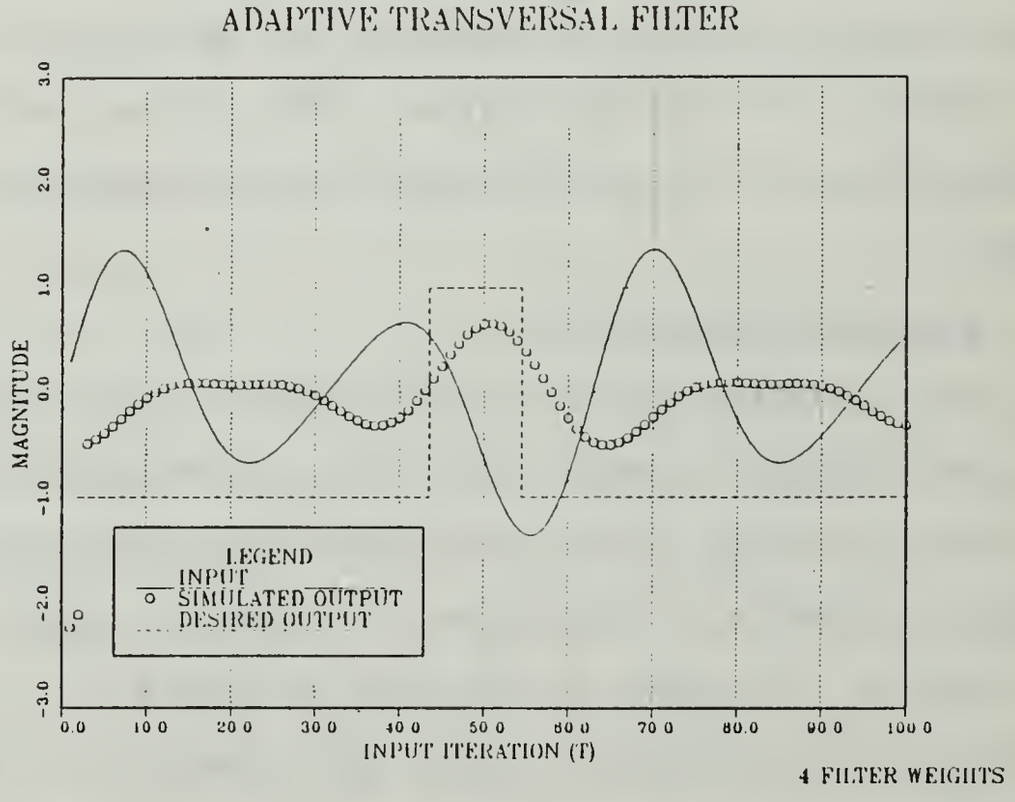


Figure 6.3 An Adaptive Transversal Filter Simulation

the three fundamental frequencies mentioned previously. The desired output (shown by the dashed line) is chosen to be a continuous -1 unless the signal of interest is detected. In that case the desired output jumps to +1.

As indicated in Figure 6.2 the Adaptive Algorithm samples the input signal and uses the successive present and three previous samples to arrive at the desired filter weights which will accomplish the task of signal discrimination. In the algorithm implemented by FIR4 of Appendix J we arrived at the following filter weights.

$$w_1 = -7.5060358$$
$$w_2 = 7.5403662$$
$$w_3 = 4.7097464$$
$$w_4 = -5.3987589$$

In Figure 6.3 we see that over 100 sample output iterations these weights resulted in an output which was at or near zero or below with a significant rise above 0.5 near the desired region. This approximates the filter response which would allow a useful discrimination of the desired signal by detection above a threshold floor (say 0.5 in this example).

This is by no means intended to be an exhaustive discussion of this approach to a matched filter design, but merely a consideration of an alternative approach which might be taken to realize a useful filter.

B. AN ANALOG SPEECH PROCESSING SCHEME

A further alternative which may be considered involves the use of commercially available speech processing microcircuits which often use Linear Predictive Coding schemes as the basis for their discriminant filters.

The current state of the art in speech recognition technology does not permit even the most sophisticated (and large) devices to recognize but several hundred words of vocabulary. The breakthroughs are most often in the arena of overcoming the speaker dependent nature of the simpler systems. However, all systems, be they single chip processors or multi-cabinet devices, do have the capability to analyze an audio input signal (conventionally this is speech of course) and to characterize the nature of changes in the formant composition over time.

Because the signal we wish to identify is in the audio spectrum it seems a logical idea to consider that a speech recognition device may prove usable for our purposes. In fact, Interstate Electronics Corporation now markets a single chip voice recognition device (VRC008) which is capable of reliable and independent recognition of up to eight words or phrases which are stored in its vocabulary. While this may seem a minimal vocabulary it is a remarkable ability for a single chip device.

The Interstate VRC008 is capable of being trained to recognize words or phrases of up to 1.2 seconds duration.

To implement an audio signal recognition scheme would require that we somehow "sample" our input audio environment in discrete blocks of about one second apiece. Thus we would simulate a discrete utterance which could be processed by the circuit. In the absence of the characteristic APU signal the device would register no recognition of the input sample. But after APU ignition it is reasonable to assume that the device would treat the APU signal as a recognizable "word" it had previously been taught.

While this approach may seem at first to be a promising one we must also consider the drawbacks, especially in view of the technically simpler approaches we have reviewed so far. First is the cost. Although the Interstate chip is by itself a relatively inexpensive device (on the order of ten dollars in quantity), by itself it is also useless without a twenty-five thousand dollar training and development tool. The intent of the manufacturer is that the cost of the development tool will be amortized by the consumer over a sizable run of usable end devices. Our application does not lend itself to mass production and the cost therefore becomes prohibitive. This is especially true in view of the cost of the simpler technologies discussed in previous chapters which have given us useful designs.

Another drawback of a speech recognition approach is the level of technical complexity versus guarantee of successful results. Unless we use a device of modular circuit size or

smaller we risk an excess of power and space consumption. But the state of the art in speech recognition is such that accuracy of recognition is roughly related to the size of the device (although it is directly the vocabulary size which is the truly overwhelming factor here). The Interstate VRC008 claims only an 85 percent accuracy of recognition which is low by the standards of other speech recognition devices. This is the price one pays for small size.

The important point is that our signal of interest is characterized by extremely well-defined and stationary spectral components. This fact allows the use of cheaper and more traditional methods of signal processing and filter design. Were our signal of a rapidly time varying nature then a purely analog approach would be impossible, and even a digital approach would prove difficult if not infeasible. It is then that methods of linear predictive coding which form the basis of speech recognition would become one of a very few viable alternatives.

VII. CONCLUSION

In this thesis I have considered several approaches to the problem of designing a matched filter for the detection of the acoustic signal which characterizes the Shuttle Auxiliary Power Unit. The Analog and Digital IIR filter approaches were treated in some detail, while the Weiner FIR and Voice Recognition methods were given less attention. My purpose was not to present an exhaustive treatise on the subject of filter design, but rather to describe various ways in which a particular problem might be approached.

It is not coincidental that the order of presentation of the considered methods should conform to the chronological introduction of these sciences to the engineering community. As may be expected, the facility with which each of these methods is employed is proportional to their general familiarity among engineers. The analog approach considered first is the best established method of filter design. Not surprisingly, this method is supported by a wealth of literature. Despite this ample documentation, at best the analog approach to filter design is an inexact science which is largely dependent upon the degree to which one is able to characterize the signals we wish to manipulate. Often, however, we have excellent knowledge of these signals, and thus the analog approach to filter design remains a

completely reasonable and certainly cost effective approach to simple filter designs.

The APU signal of concern to this study was such a signal. Its signature was stationary over time and could be reliably found at amplitudes well above the noise threshold. The dominant component at 600 Hz was of quality sufficient to preclude examination of sub-dominant spectral harmonics at higher frequencies. The fact that a well-defined signal was evident allowed for a design which emphasized the simplicity of the analog approach.

Mention should be made of the obstacles which did impede the final analog design. Because an analog filter serves only to attenuate those signal component spectral elements out of the passband, but does not eliminate them, it is necessary to know the range of amplitude which may be expected of the sensor microphone output. For a given amplitude of signal input which varies little within the range of input frequencies it may be reliably expected that the analog bandpass filter would reject the frequency components outside of the narrow passband. But if the spectral components were grossly disparate in their amplitude and a component out of the passband were received which was significantly above the amplitude expected of the 600 Hz center frequency, then it is possible that the component out of the passband would be passed regardless of

the filter attenuation. This demonstrates the need we have to know the nature of the input signal.

One approach to this problem is to increase the attenuation of the filter. But this does not guarantee signal component rejection out of the passband. The solution for an analog approach lies in Automatic Gain Control (AGC) at the sensor microphone input to the filter. In this manner we can ensure a dynamic range of input signal which is within the limits of filter discrimination. This implies a careful selection of a microphone and preamplifier combination which in turn implies a similarly careful understanding of the dynamic range of the input signal. Indeed, these considerations continue to be the most vexing aspects of a useful final design. The actual dynamic range of the signals recorded on tape was unreliable due to the number of intermediate and indeterminate dubs which the tape underwent prior to our acquisition of a copy.

Furthermore, and even more importantly, the ultimate placement of the sensor microphone in the Shuttle cargo bay will have considerable effect upon the nature of the signal available for discrimination. It will also tell significantly on the dynamic range. This factor will impact upon any chosen filter design regardless of the algorithm selected. Thus in the analog case we must design for a wide dynamic range and provide AGC which yields a narrower range of amplitude input into the filter.

Much of this problem is overcome with the digital filter implementation developed in Chapters 4 and 5. At the foundation of the digital design is a frequency domain scheme whose output is less dependent upon input amplitude variations than the frequency components of the input signal. In fact using an EPROM based filter design such as afforded by the INTEL 2920 we enjoy considerable flexibility in tailoring the range of allowable inputs and outputs through careful selection of program parameters. The limitations are rather imposed by the noise level at the low end and the power limit at the high end.

There are several drawbacks to the digital filter which bear mentioning. The foremost drawback is cost relative to the analog filter. The design presented in Chapter 5 was dependent upon the SDK-2920 Development Kit which is a thousand dollar item. This is the minimum hardware which is necessary to develop a 2920 signal processing design. However, to support any sort of a sophisticated development requires the INTEL Intellec Development System with associated software. This quickly elevates the expense of the system to a range of tens of thousands. Of course there are certainly more uses for the Intellec system than simply a 2920 development application, so this expense can be amortized over those additional uses. But the 2920 applications software which supports the Intellec system is a four thousand dollar expense by itself.

This fact proved significant to the digital design when the simulator software was found to have a bug in it. When the original disks could not be located it was then deemed more practical to develop an application specific simulation on a mainframe computer instead of purchasing a replacement package from INTEL. This meant an additional expenditure of time of course, and was only successful in showing the adequacy of the specific 2920 filter implementation algorithm. However, without the 2920 Simulator software package effective troubleshooting was made significantly more difficult. Nontheless, as indicated in the results of Chapter 5, a successful 2920 implementation was accomplished without a fully healthy simulator. With it the design process would have been considerably more efficient.

An additional consideration is the complexity of the digital design over the analog approach. This is due in large part to the availability of resources which support an analog design relative to the novel approach represented by a state-of-the-art signal processing chip. However, the complexity of a signal processor application is often far outweighed by the considerable flexibility which it provides. One must not forget the power of the 2920 (witness its ability to incorporate all of the hardware elements of the follow-on logic circuitry required in the analog design in but a dozen or so lines of 2920 assembly language code) and weigh this against the short term

inconvenience of having to become acquainted with a new approach. Once mastered the significant ability of a signal processing device make an analog approach to any complex filter design seem archaic. In addition the fewer actual circuit components required in an EPROM based device means significant savings in power consumption. This is an especially noteworthy item when considering an electronic device for a space application.

For the purposes of this thesis I must admit that the 2920 was certainly fun to work with. The literature is sketchy in spots and several calls to technical support at INTEL were needed to resolve some issues and errors. But overall the 2920 certainly provides the researcher with a significant amount of flexible and powerful signal processing ability.

The significant advantage of implementing a digital filter over the analog design is the relative immunity to the variations in input amplitude. This was a crucial consideration in the development described in Chapter 5 and by itself would account for the choice of the digital design over the analog approach. When coupled with the further advantages of lower power consumption, less physical space required and considerable flexibility in accommodating future changes without the need for hardware modification, the digital approach implemented in a powerful signal processor becomes an irresistable filter design option.

In Chapter 6 we considered two other approaches to the APU signal detection problem. Unfortunately a lack of time prohibited a serious examination of these additional approaches. Both are well-founded and represent the leading edge of signal processing technology. Given a requirement for detection of a more complex signal than we considered in this paper, these latter methodological options could well represent the only viable means of processing a time-varying signal in the acoustic spectrum.

APPENDIX A

QUICK LOOK DATA PACKET

TEST TITLE APU HOTFIRE STS-2

TEST DATE 15 SEPTEMBER 81

RUN NO. (S)

REDUCTION DATE 18 SEPTEMBER 81

FACILITY JSC NASA VATF BLDG #49

SEQUENCE NO. (S)

DATA TYPE PSD PLOTS D0280A and D0281A + Noise Floor APU 2 x-axis APU 2 Z

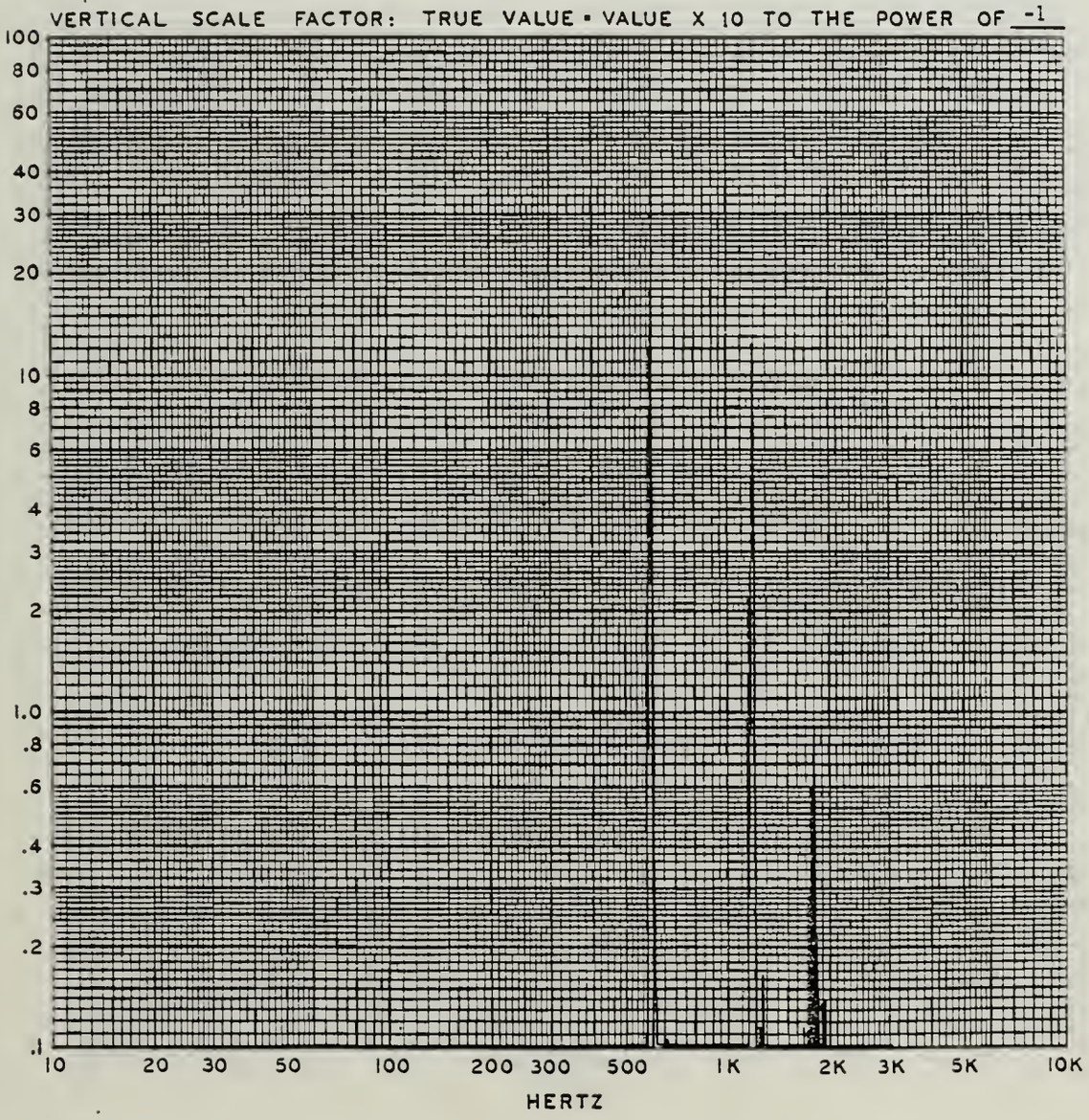
NASA-JSC

NSI-SML FORM 147 (MAY. 79)

RUN IDENTIFICATION APU HOTFIRE CHANNEL 11 STS-2

PARAMETER D0280A	TEST DATE 15 September 81
OVERALL RMS VALUE 6.4863 G's	ANALYSIS B W 2.2467 Hz
ENGINEERING UNITS 30.0000 G's	ANALOG L.P. FILTER B W 2300.0000 Hz
START TIME: (HR: MIN: SEC) 258/11:56:35	DEGREES OF FREEDOM 9.0000
SAMPLE RATE (S/SEC) 6134.9687	PROCESS DATE 18 September 81

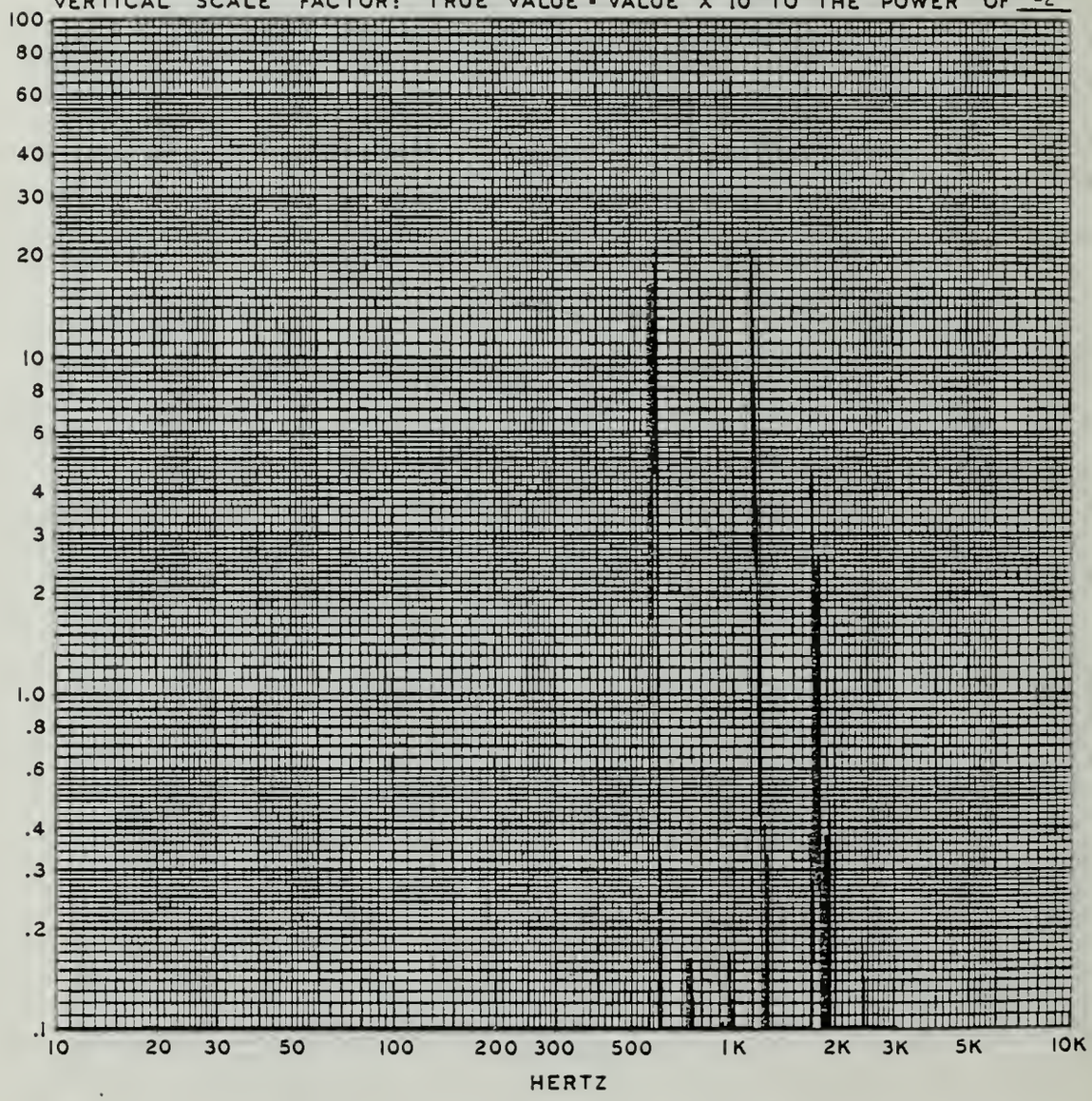
VERTICAL SCALE FACTOR: TRUE VALUE = VALUE X 10 TO THE POWER OF -1



RUN IDENTIFICATION APU HOTFIRE CHANNEL 11 STS-2

PARAMETER 00280A	TEST DATE 15 September 81
OVERALL RMS VALUE 3.1781 G's	ANALYSIS B W 2.2467 Hz
ENGINEERING UNITS 30.0000 G's	ANALOG L.P. FILTER B W 2300.0000 Hz
START TIME: (HR: MIN: SEC) 258/11:57:46	DEGREES OF FREEDOM 9.0000
SAMPLE RATE (S/SEC) 6134.9687	PROCESS DATE 18 September 81

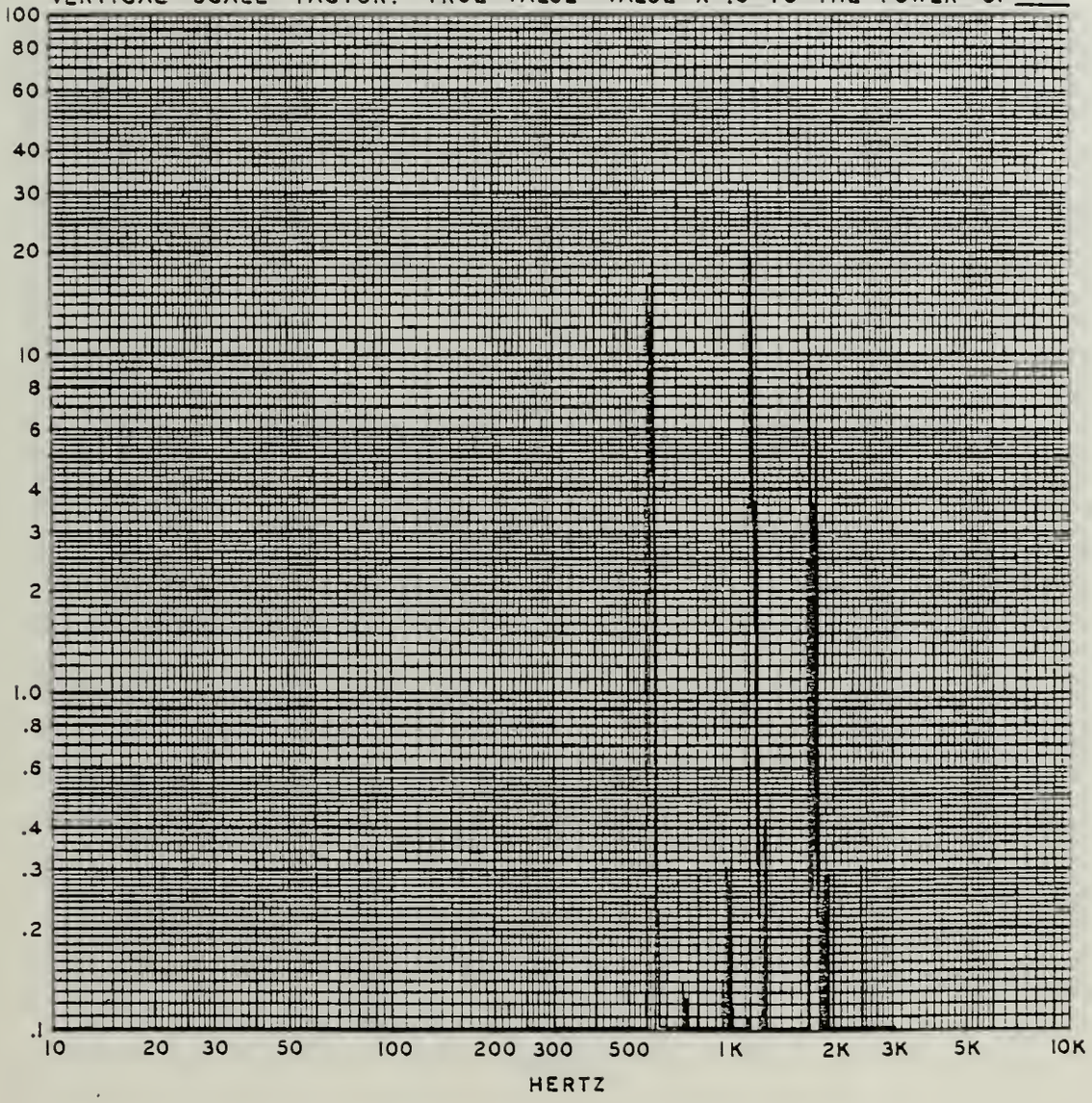
VERTICAL SCALE FACTOR: TRUE VALUE = VALUE X 10 TO THE POWER OF -2



RUN IDENTIFICATION APU HOTFIRE CHANNEL 11 STS-2

PARAMETER D0280A	TEST DATE 15 September 81
OVERALL RMS VALUE 3.6335 G's	ANALYSIS B W 2.2467 Hz
ENGINEERING UNITS 30.0000 G's	ANALOG L.P. FILTER B W 2300.0000 Hz
START TIME: (HR: MIN: SEC) 258/11:58:14	DEGREES OF FREEDOM 9.0000
SAMPLE RATE (S/SEC) 6134.9687	PROCESS DATE 18 September 81

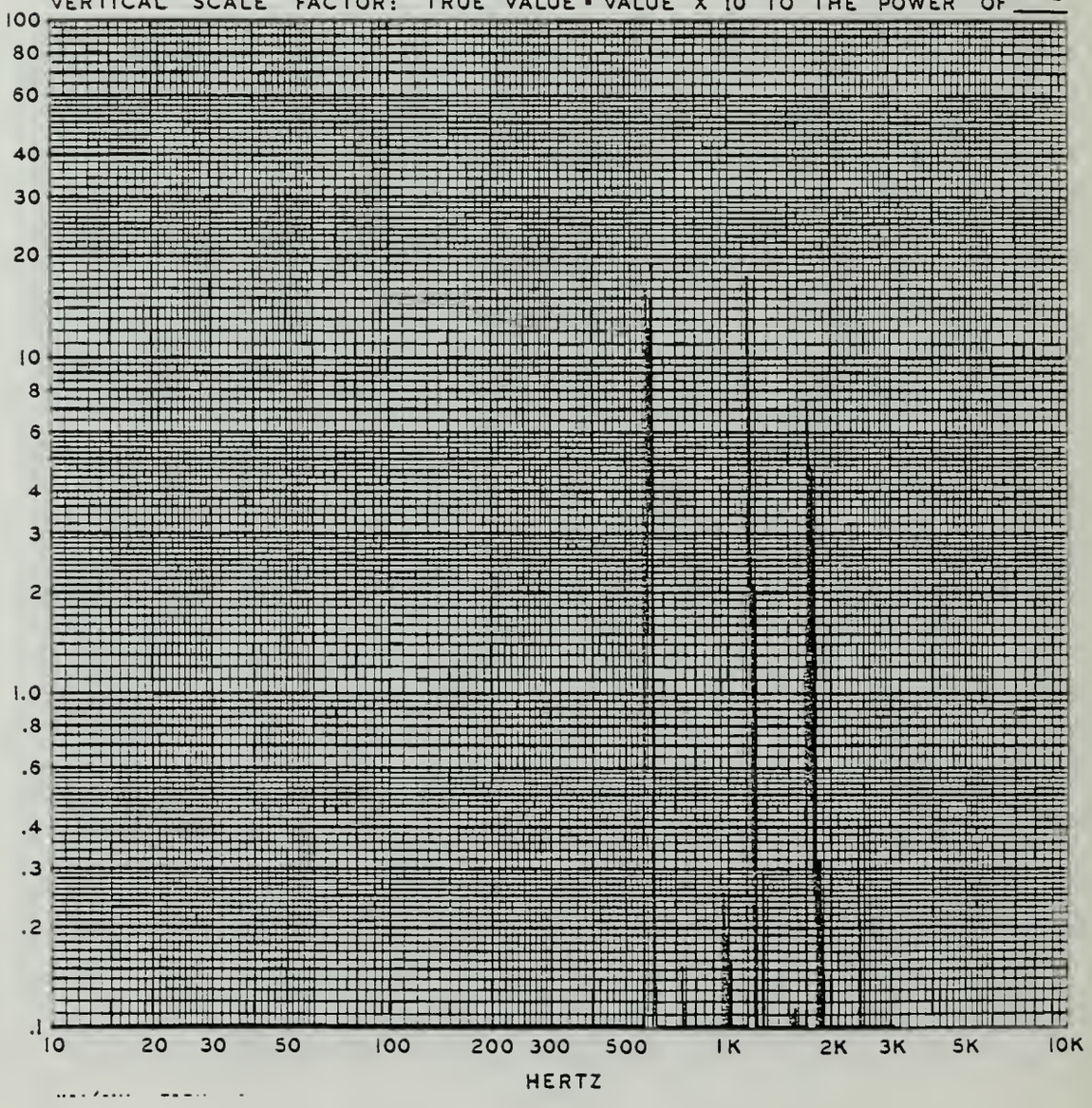
VERTICAL SCALE FACTOR: TRUE VALUE = VALUE X 10 TO THE POWER OF -2



RUN IDENTIFICATION APU HOTFIRE CHANNEL 11 STS-2

PARAMETER D0280A	TEST DATE 15 September 81
OVERALL RMS VALUE 2.9619 G's	ANALYSIS B W 2.2467 Hz
ENGINEERING UNITS 30.0000 G's	ANALOG L.P. FILTER B W 2300.0000 Hz
START TIME: (HR: MIN: SEC) 258/12:01:22	DEGREES OF FREEDOM 9.0000
SAMPLE RATE (S/SEC) 6134.9687	PROCESS DATE 18 September 81

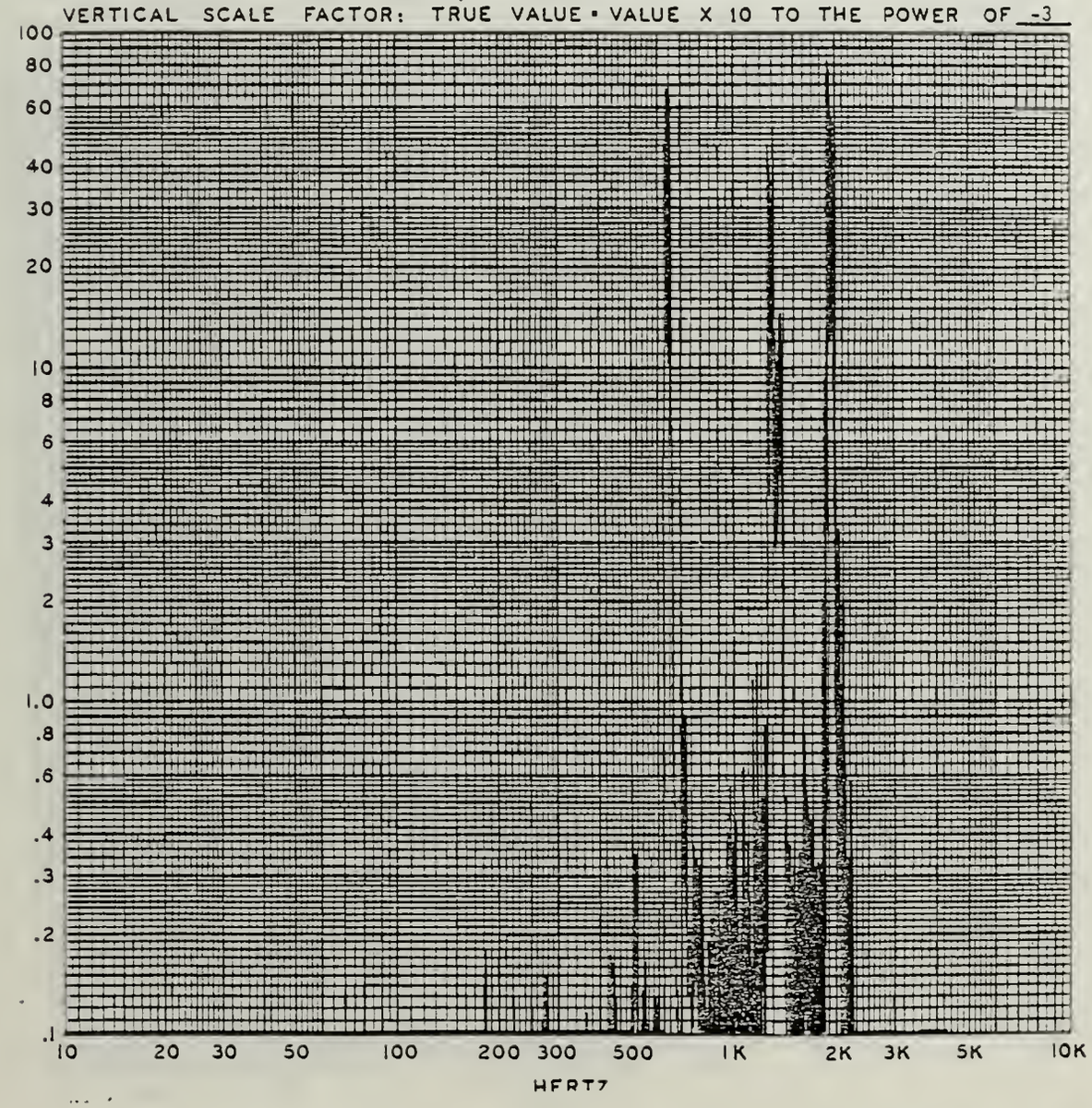
VERTICAL SCALE FACTOR: TRUE VALUE = VALUE X 10 TO THE POWER OF -2



RUN IDENTIFICATION APU HOTFIRE CHANNEL 11 STS-2

PARAMETER D0280A	TEST DATE 15 September 81
OVERALL RMS VALUE 2.7269 G's	ANALYSIS B W 2.2467 Hz
ENGINEERING UNITS 30.0000 G's	ANALOG L.P. FILTER B W 2300.0000 Hz
START TIME: (HR: MIN: SEC) 258/12:03:20	DEGREES OF FREEDOM 9.0000
SAMPLE RATE (S/SEC) 6134.9687	PROCESS DATE 18 September 81

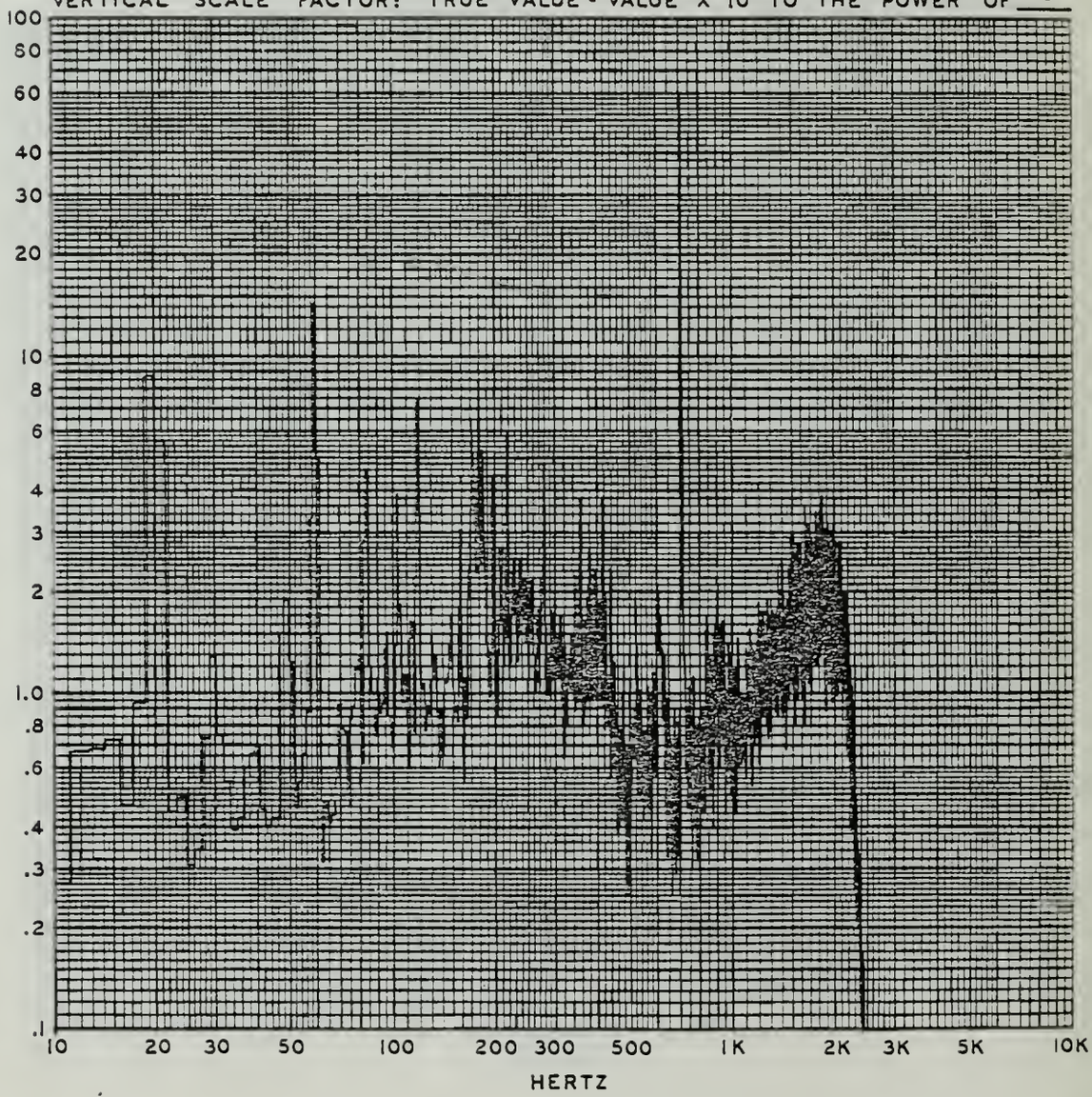
VERTICAL SCALE FACTOR: TRUE VALUE = VALUE X 10 TO THE POWER OF -3



RUN IDENTIFICATION APU HOTFIRE CHANNEL 11 STS-2

PARAMETER	D0280A Noise Floor	TEST DATE	15 September 81
OVERALL RMS VALUE	0.1920 G's	ANALYSIS B W	2.2467 Hz
ENGINEERING UNITS	30.0000 Gs	ANALOG L.P. FILTER B W	2300.0000 Hz
START TIME: (HR: MIN: SEC)	258/12:04:20	DEGREES OF FREEDOM	9.0000
SAMPLE RATE (S/SEC)	6134.9687	PROCESS DATE	18 September 81

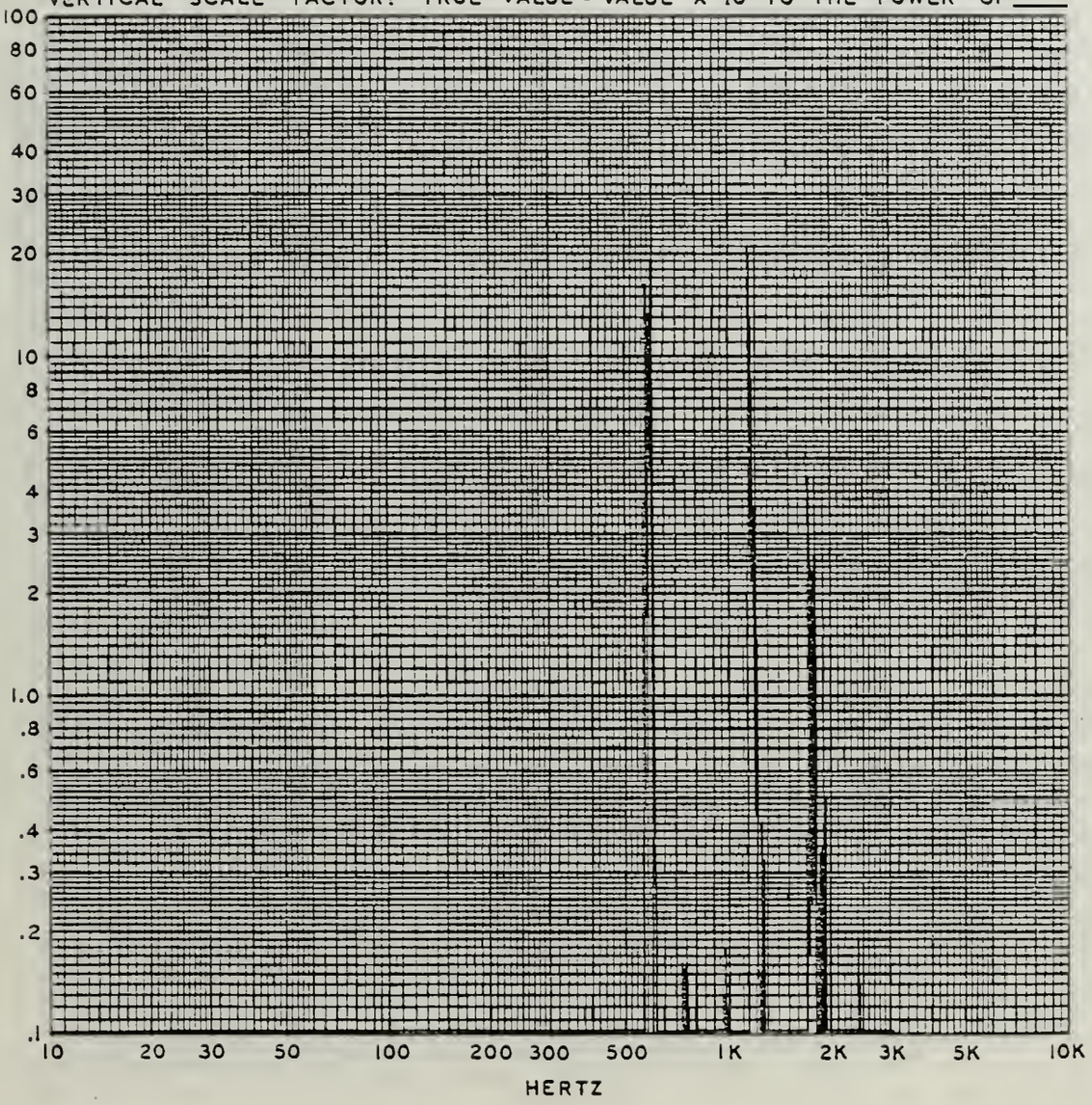
VERTICAL SCALE FACTOR: TRUE VALUE = VALUE X 10 TO THE POWER OF -5



RUN IDENTIFICATION APU HOTFIRE CHANNEL 10 STS-2

PARAMETER D0281A	TEST DATE 15 September 81
OVERALL RMS VALUE 3.1738 G's	ANALYSIS B W 2.2467 Hz
ENGINEERING UNITS 30.0000 G's	ANALOG L.P. FILTER B W 2300.0000 Hz
START TIME: (HR: MIN: SEC) 258/11:57:46	DEGREES OF FREEDOM 9.0000
SAMPLE RATE (S/SEC) 6134.9687	PROCESS DATE 18 September 81

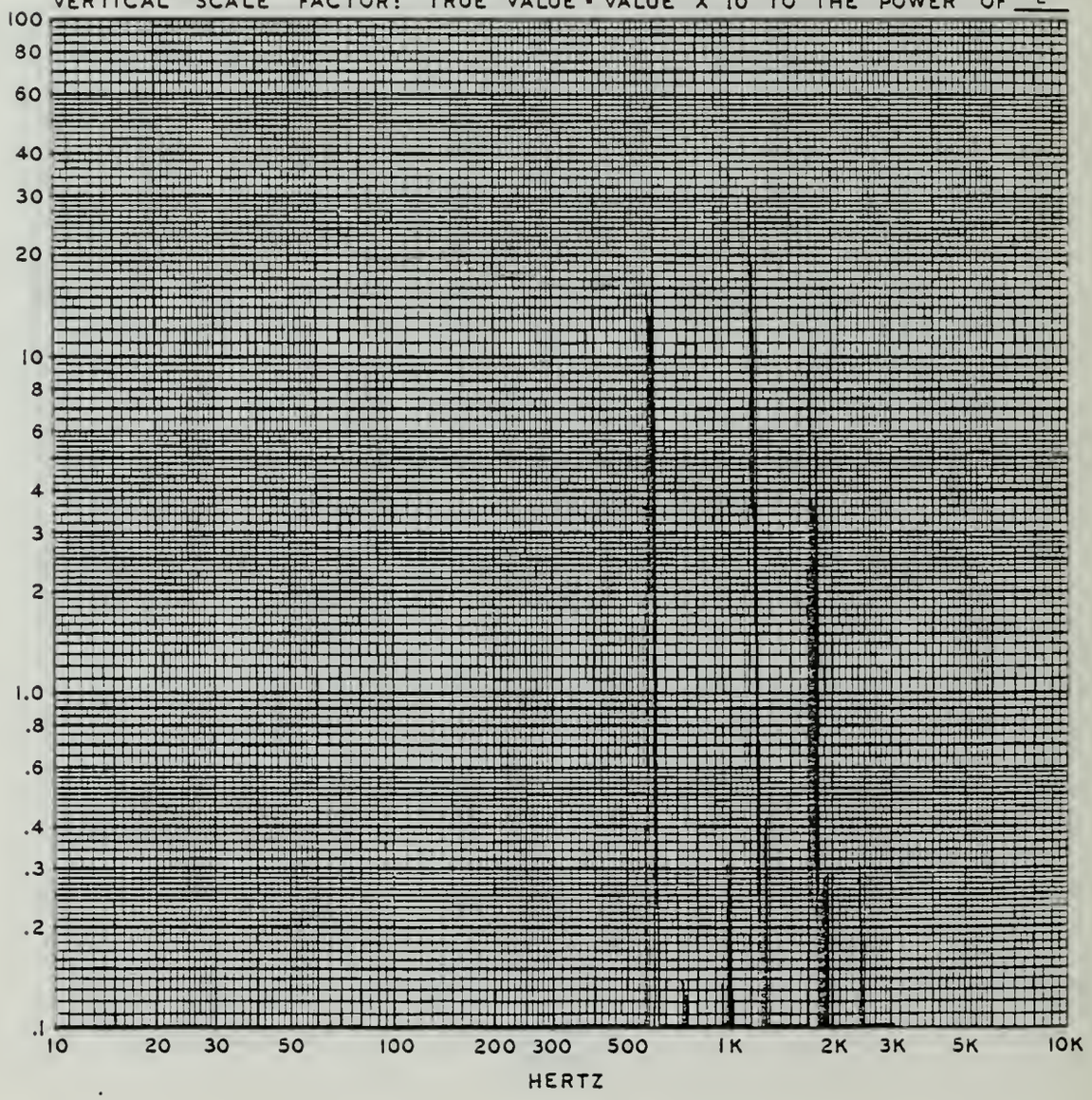
VERTICAL SCALE FACTOR: TRUE VALUE = VALUE X 10 TO THE POWER OF -2



RUN IDENTIFICATION APU HOTFIRE CHANNEL 1D STS-2

PARAMETER	D0281 A	TEST DATE	15 September 81
OVERALL RMS VALUE	3.6288 G's	ANALYSIS B W	2.2467 Hz
ENGINEERING UNITS	30.0000 G's	ANALOG L.P. FILTER B W	2300.0000 Hz
START TIME: (HR: MIN: SEC)	258/11:58:14	DEGREES OF FREEDOM	9.0000
SAMPLE RATE (S/SEC)	6134.9687	PROCESS DATE	18 September 81

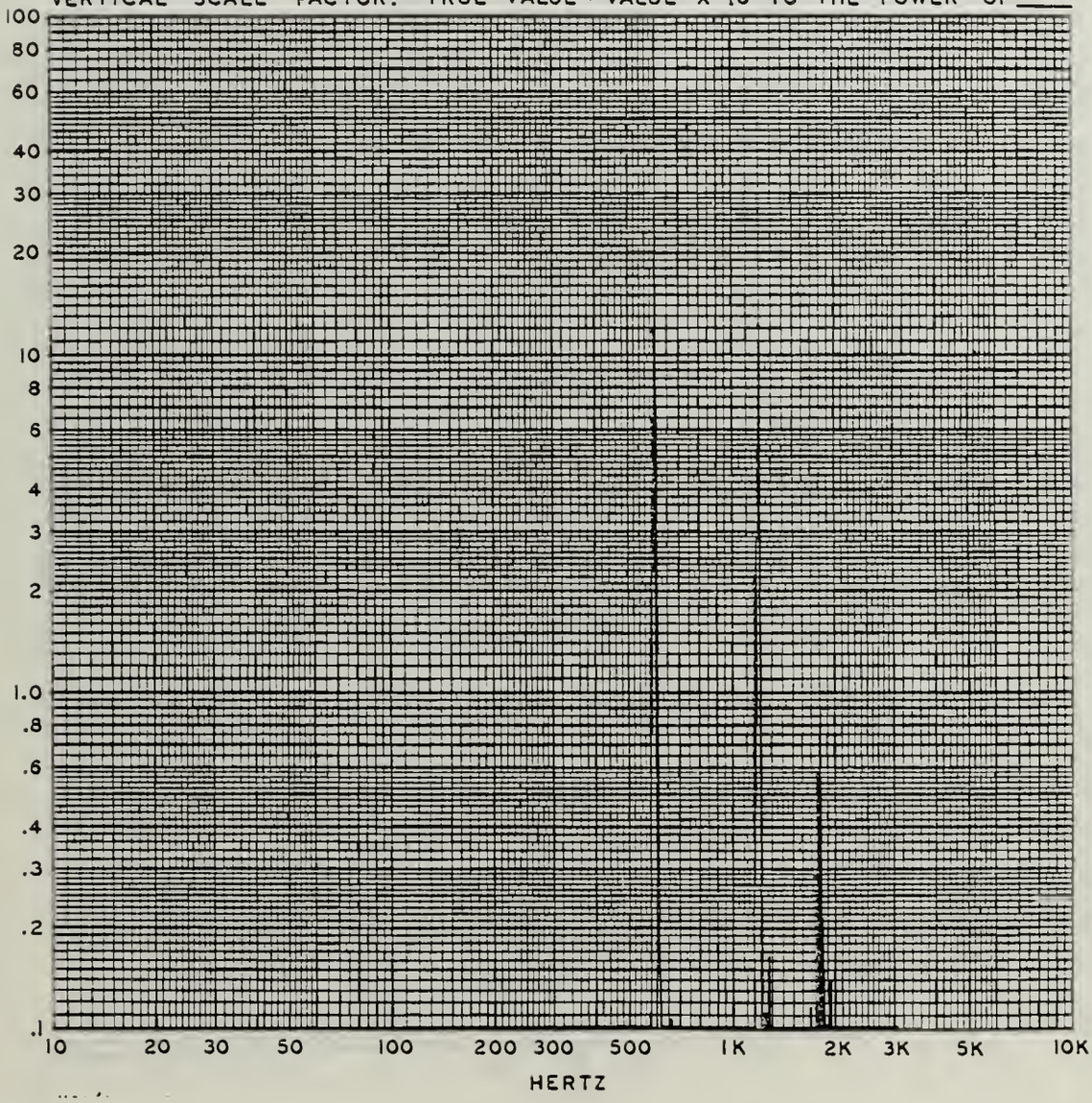
VERTICAL SCALE FACTOR: TRUE VALUE = VALUE X 10 TO THE POWER OF -2



RUN IDENTIFICATION APU HOTFIRE CHANNEL 10 STS-2

PARAMETER	D0281A	TEST DATE	15 September 81
OVERALL RMS VALUE	6.4787 G's	ANALYSIS B W	2.2467 Hz
ENGINEERING UNITS	30.0000 G's	ANALOG L.P. FILTER B W	2300.0000 Hz
START TIME: (HR: MIN: SEC)	258/11:56:35	DEGREES OF FREEDOM	9.0000
SAMPLE RATE (S/SEC)	6134.9687	PROCESS DATE	18 September 81

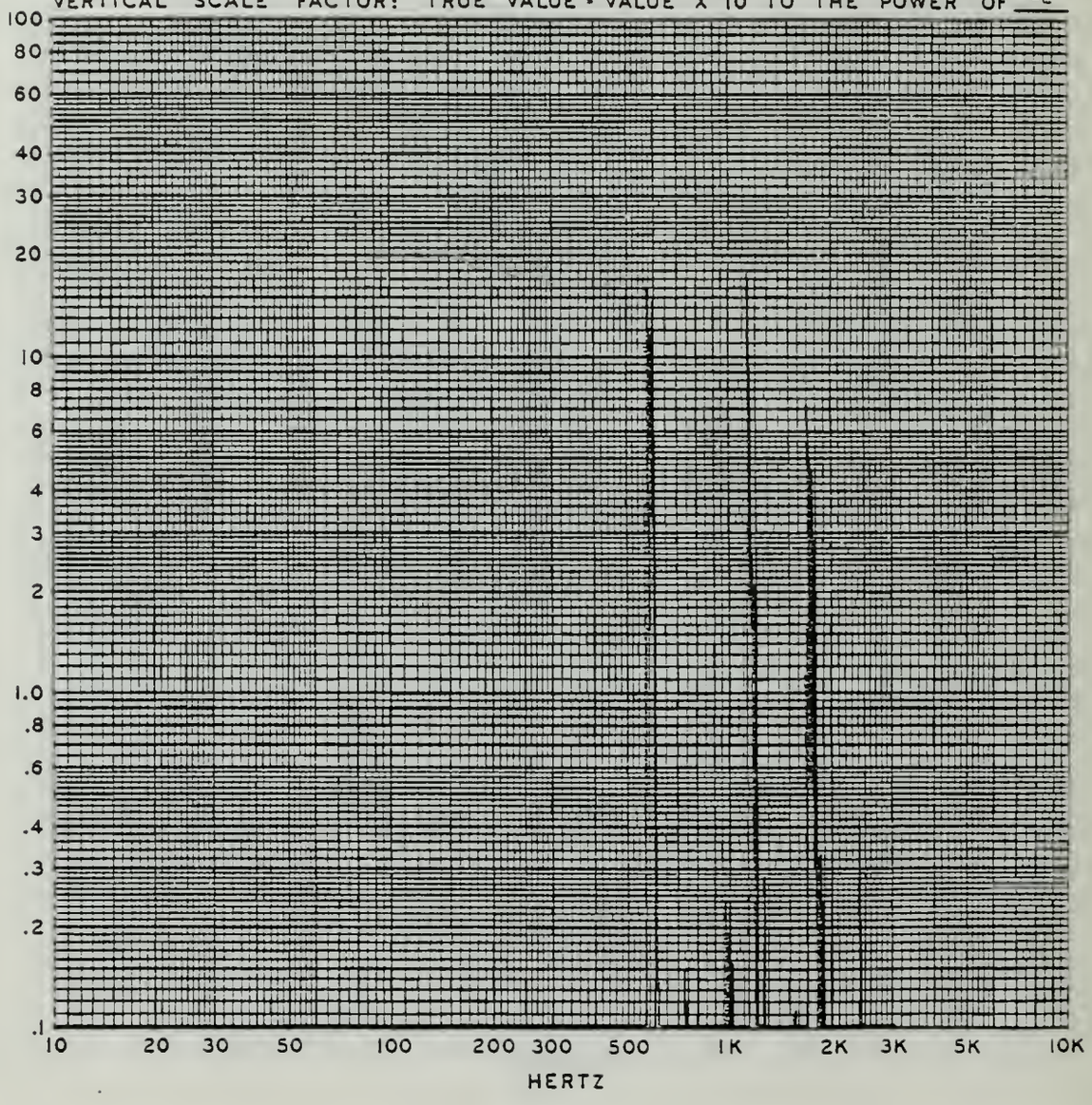
VERTICAL SCALE FACTOR: TRUE VALUE = VALUE X 10 TO THE POWER OF -1



RUN IDENTIFICATION APU HOTFIRE CHANNEL 10 STS-2

PARAMETER	D0281A	TEST DATE	15 September 81
OVERALL RMS VALUE	2.9631 G's	ANALYSIS B W	2.2467 Hz
ENGINEERING UNITS	30.0000 G's	ANALOG L.P. FILTER B W	2300.0000 Hz
START TIME: (HR: MIN: SEC)	258/12:01:22	DEGREES OF FREEDOM	9.0000
SAMPLE RATE (S/SEC)	6134.9687	PROCESS DATE	18 September 81

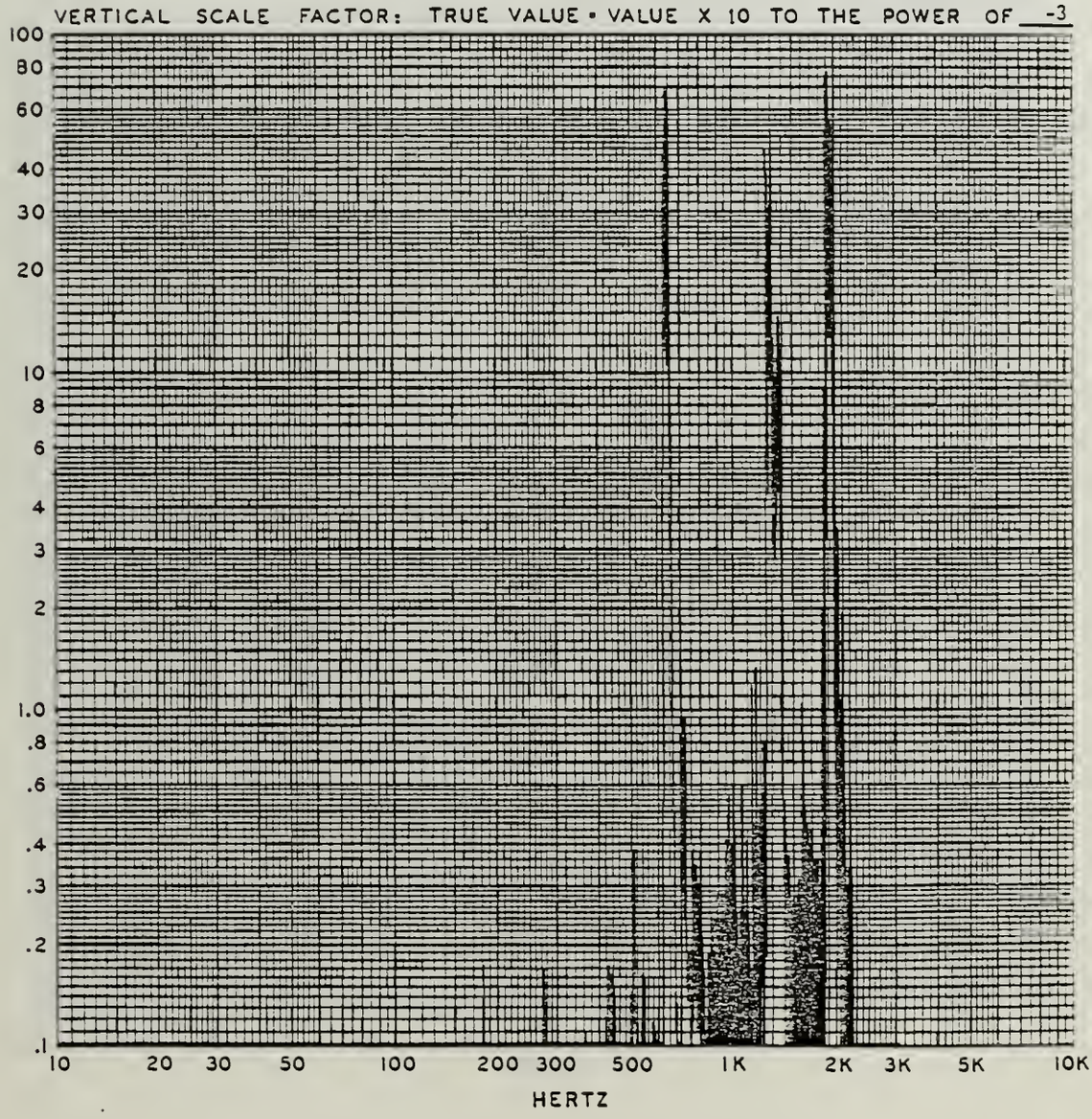
VERTICAL SCALE FACTOR: TRUE VALUE = VALUE X 10 TO THE POWER OF -2



RUN IDENTIFICATION APU HOTFIRE CHANNEL 10 STS-2

PARAMETER D0281A	TEST DATE 15 September 81
OVERALL RMS VALUE 2.7251 G's	ANALYSIS B W 2.2467 Hz
ENGINEERING UNITS 30.0000 G's	ANALOG L.P. FILTER B W 2300.0000 Hz
START TIME: (HR: MIN: SEC) 258/12:03:20	DEGREES OF FREEDOM 9.0000
SAMPLE RATE (S/SEC) 6134.9687	PROCESS DATE 18 September 81

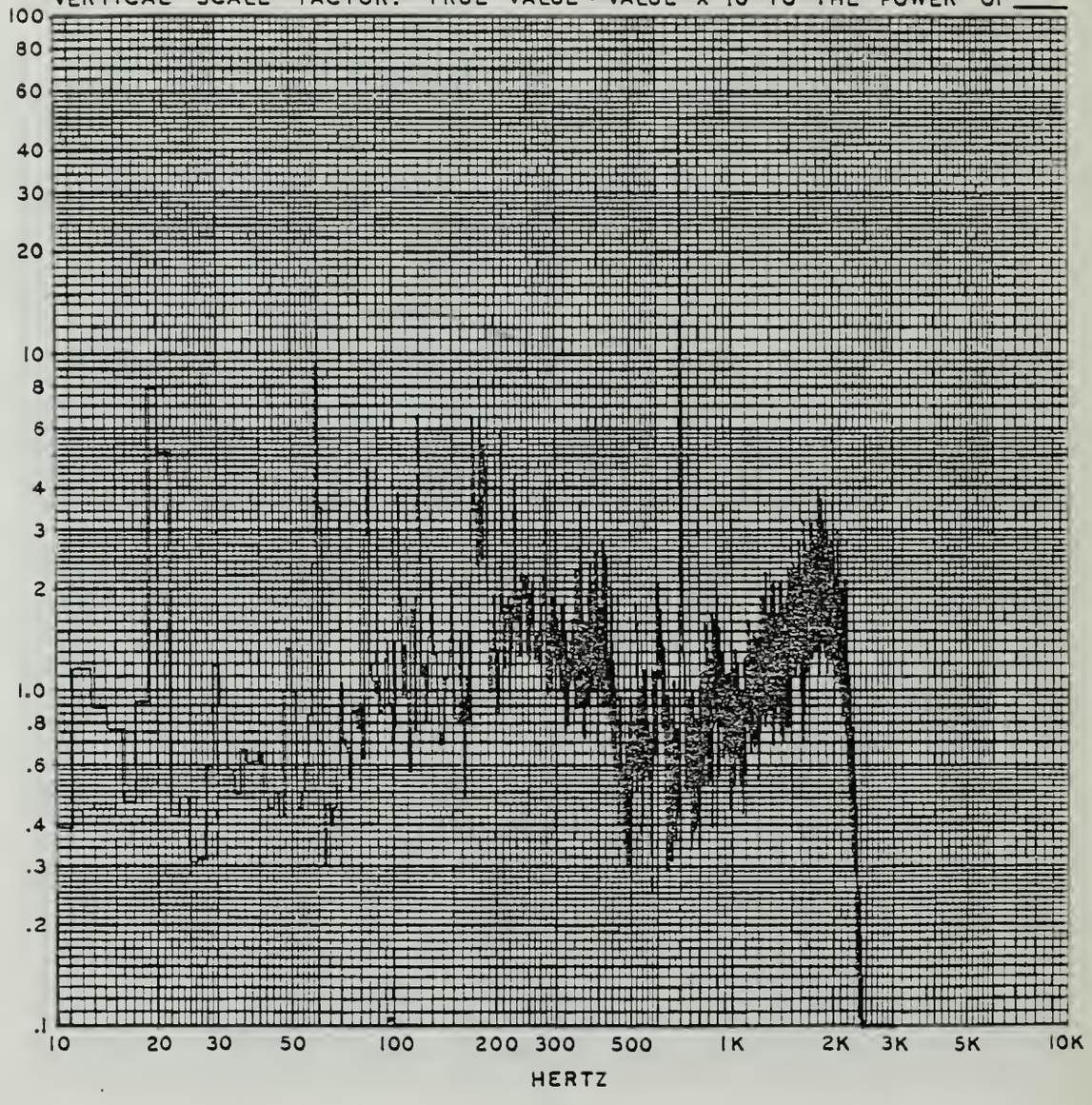
VERTICAL SCALE FACTOR: TRUE VALUE = VALUE X 10 TO THE POWER OF -3



RUN IDENTIFICATION APU HOTFIRE CHANNEL 10 STS-2

PARAMETER	D0281A Noise Floor	TEST DATE	15 September 81
OVERALL RMS VALUE	0.1905 G's	ANALYSIS B W	2.2467 Hz
ENGINEERING UNITS	30.0000 G's	ANALOG L.P. FILTER B W	2300.0000 Hz
START TIME: (HR: MIN: SEC)	258/12:04:20	DEGREES OF FREEDOM	9.0000
SAMPLE RATE (S/SEC)	6134.9687	PROCESS DATE	18 September 81

VERTICAL SCALE FACTOR: TRUE VALUE = VALUE X 10 TO THE POWER OF -5



APPENDIX B

POWER SPECTRAL DENSITY PLOTS OF THE SHUTTLE CARGO BAY PRE-LAUNCH ACOUSTIC ENVIRONMENT

Legend for the Graphical Output on the Following Pages

Shuttle Flight Number: STS-2, STS-3 or STS-4

Microphone Identification: 9405, 9219 or 9403

Sampled Interval Relative to APU Power-Up: PRE or POST

All PSD Sources are from the Original Aerospace Tape Copies (labelled ORIG)

Narrow Band Analysis (N=40 Samples)

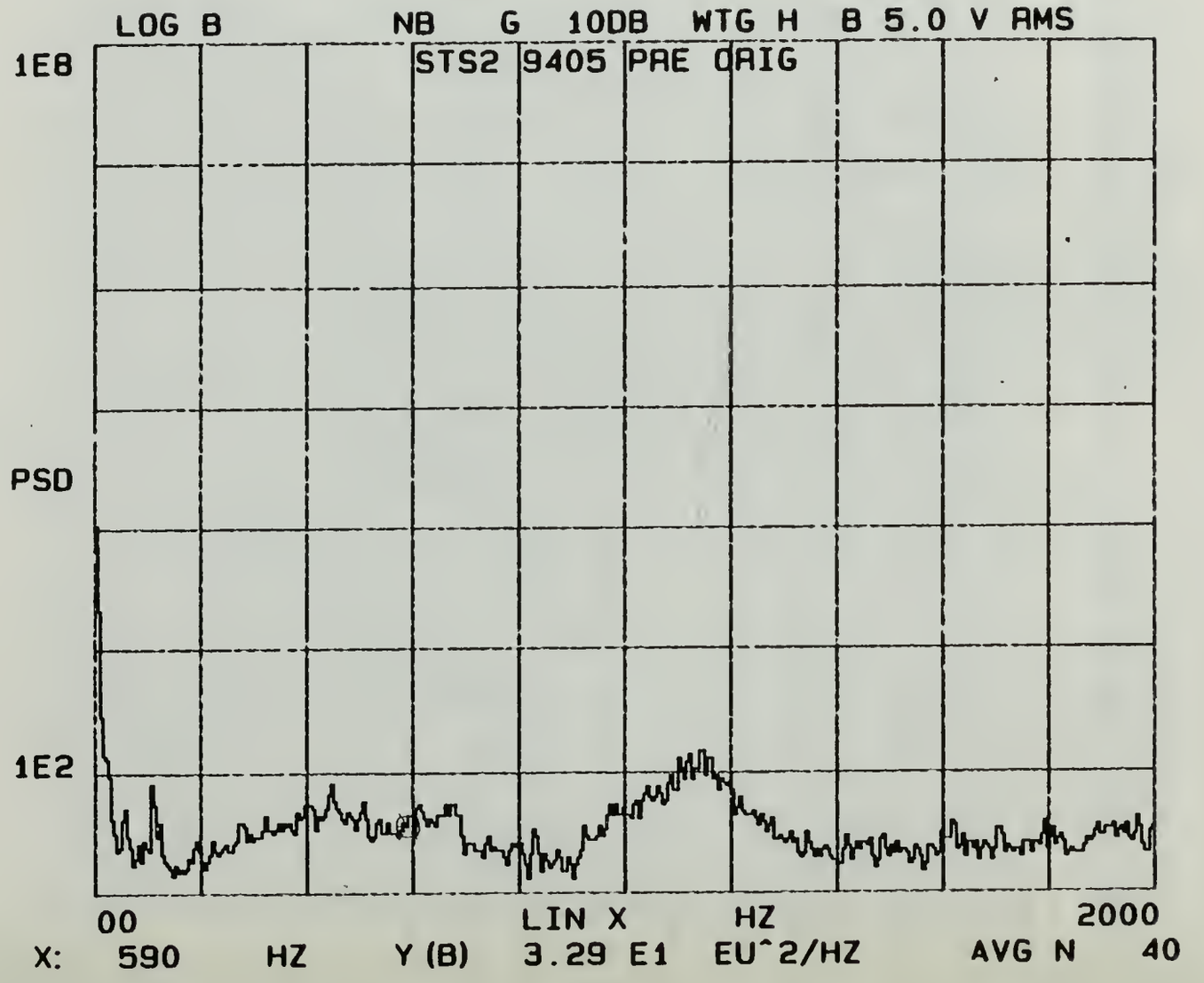
Hanning Weighting

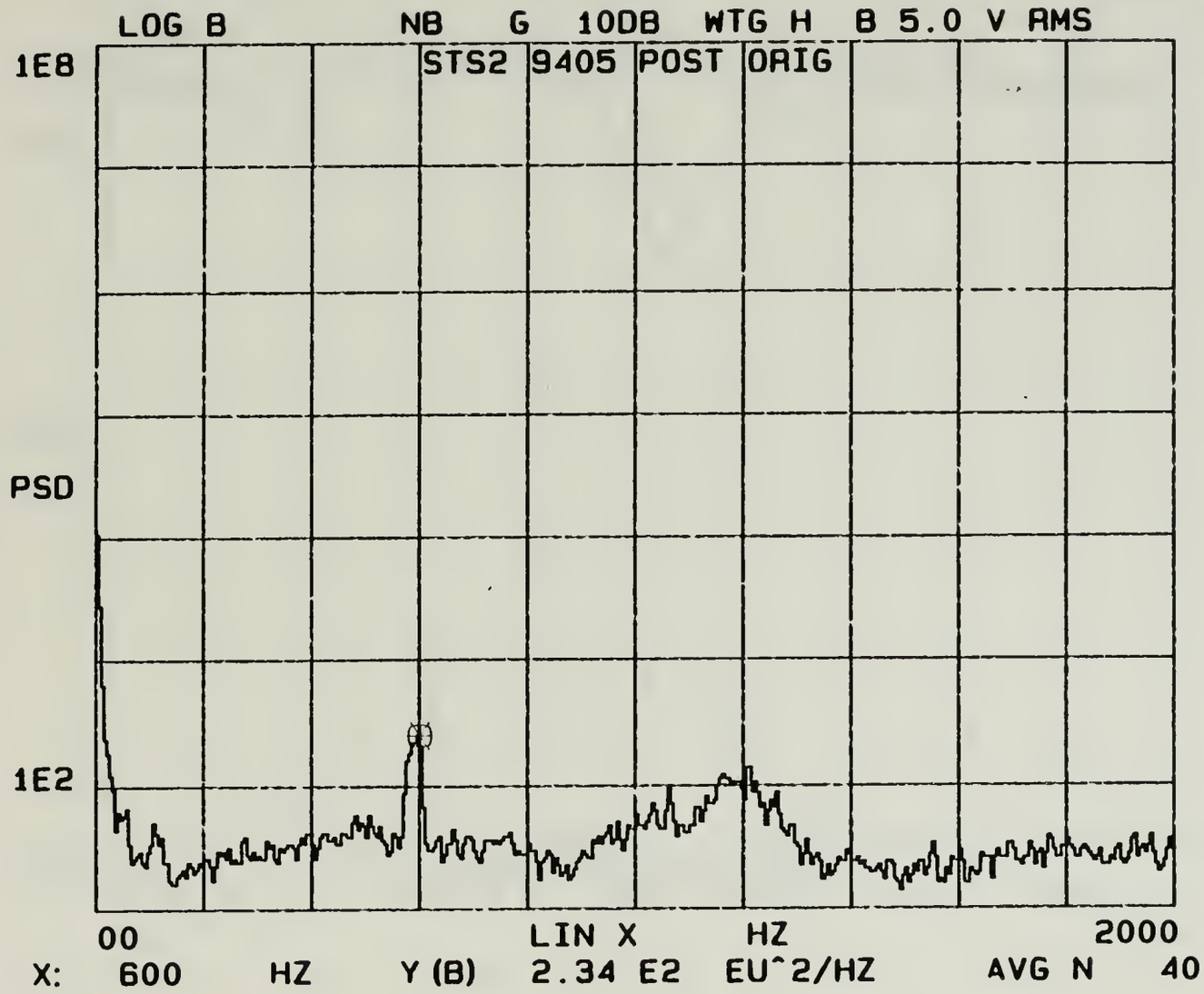
5.0 Volt RMS Front End Limiter

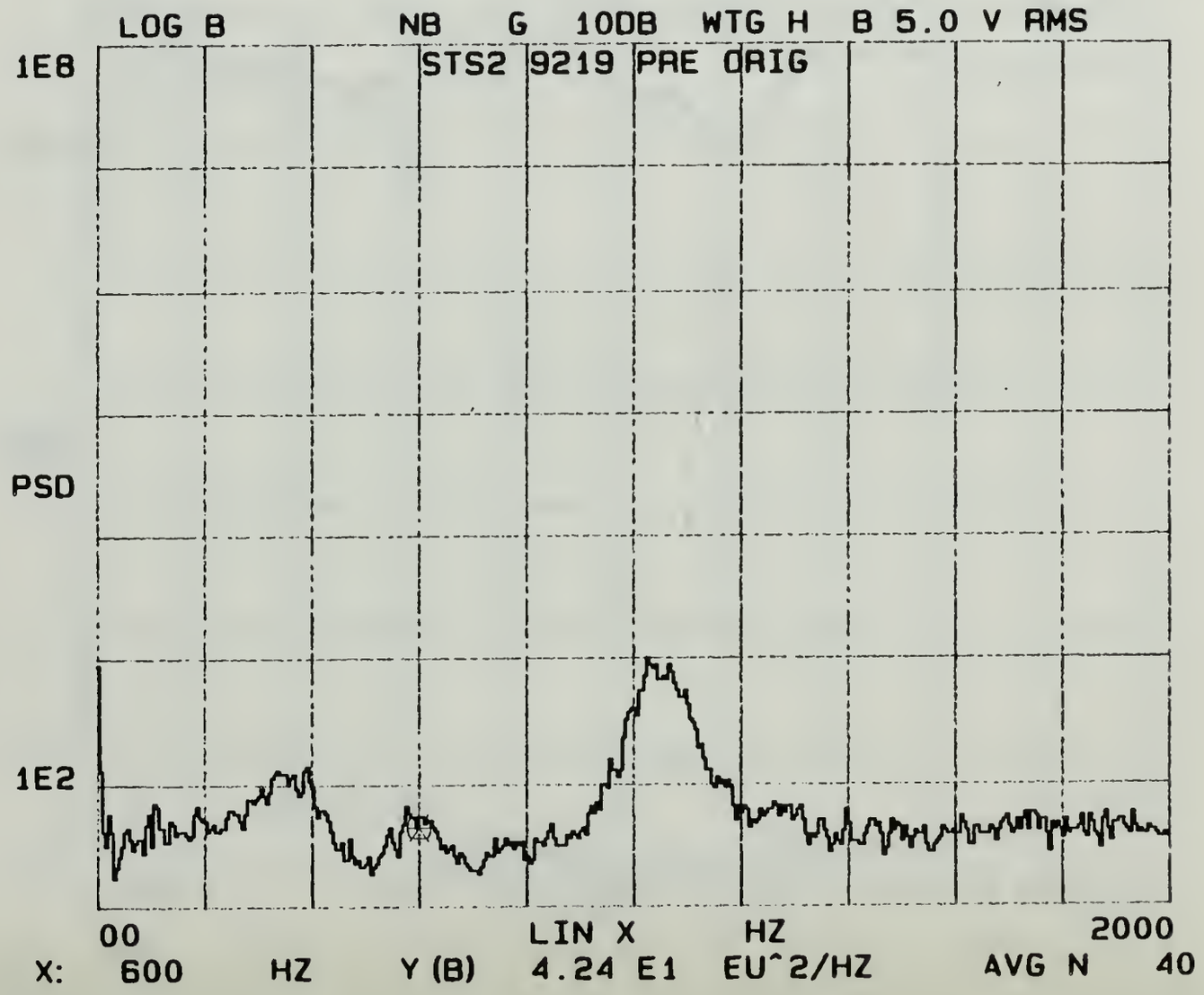
Gain: 10 dB per division

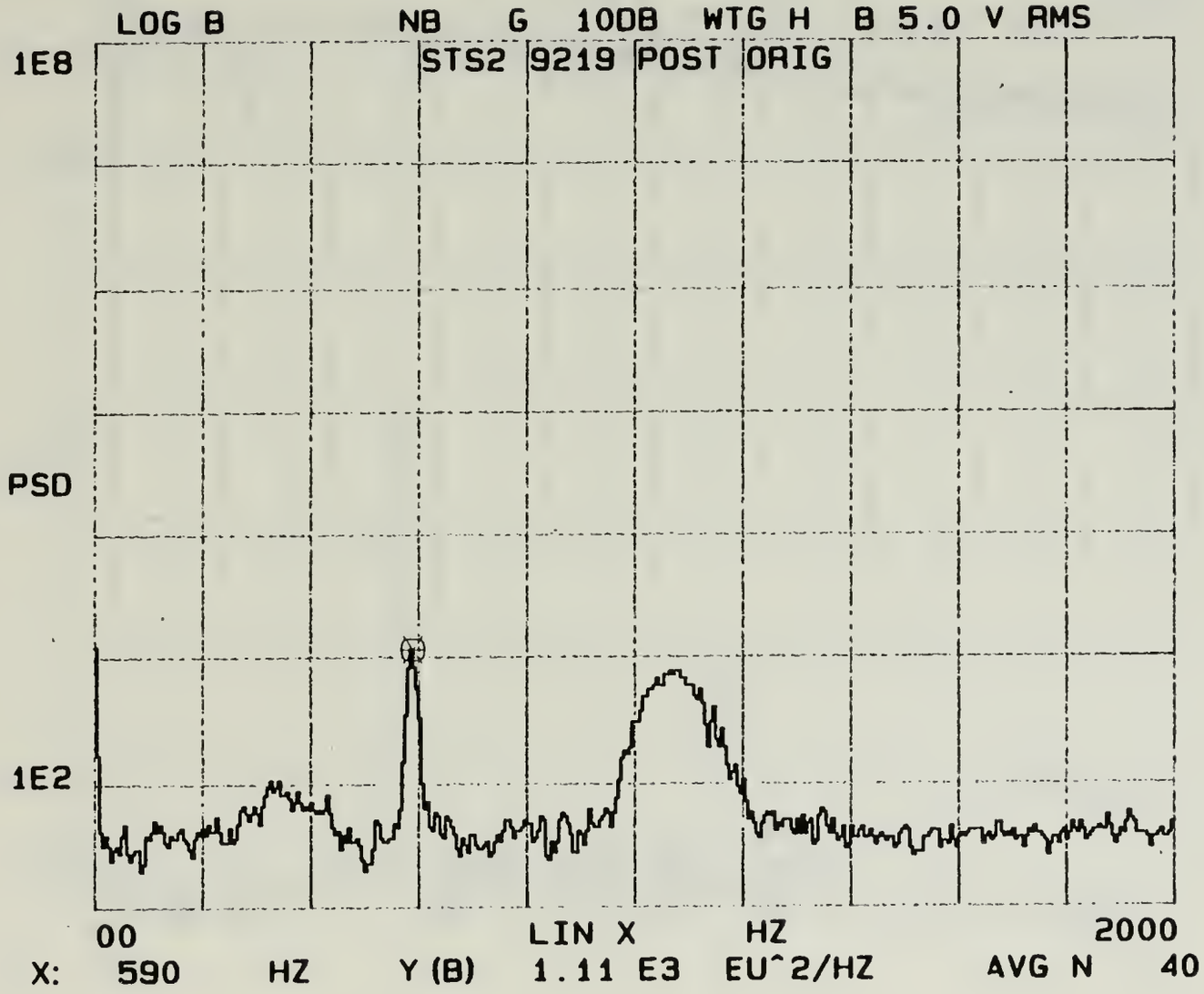
Cursor Point Label: X (Hz) and Y(B) (Engineering Units)

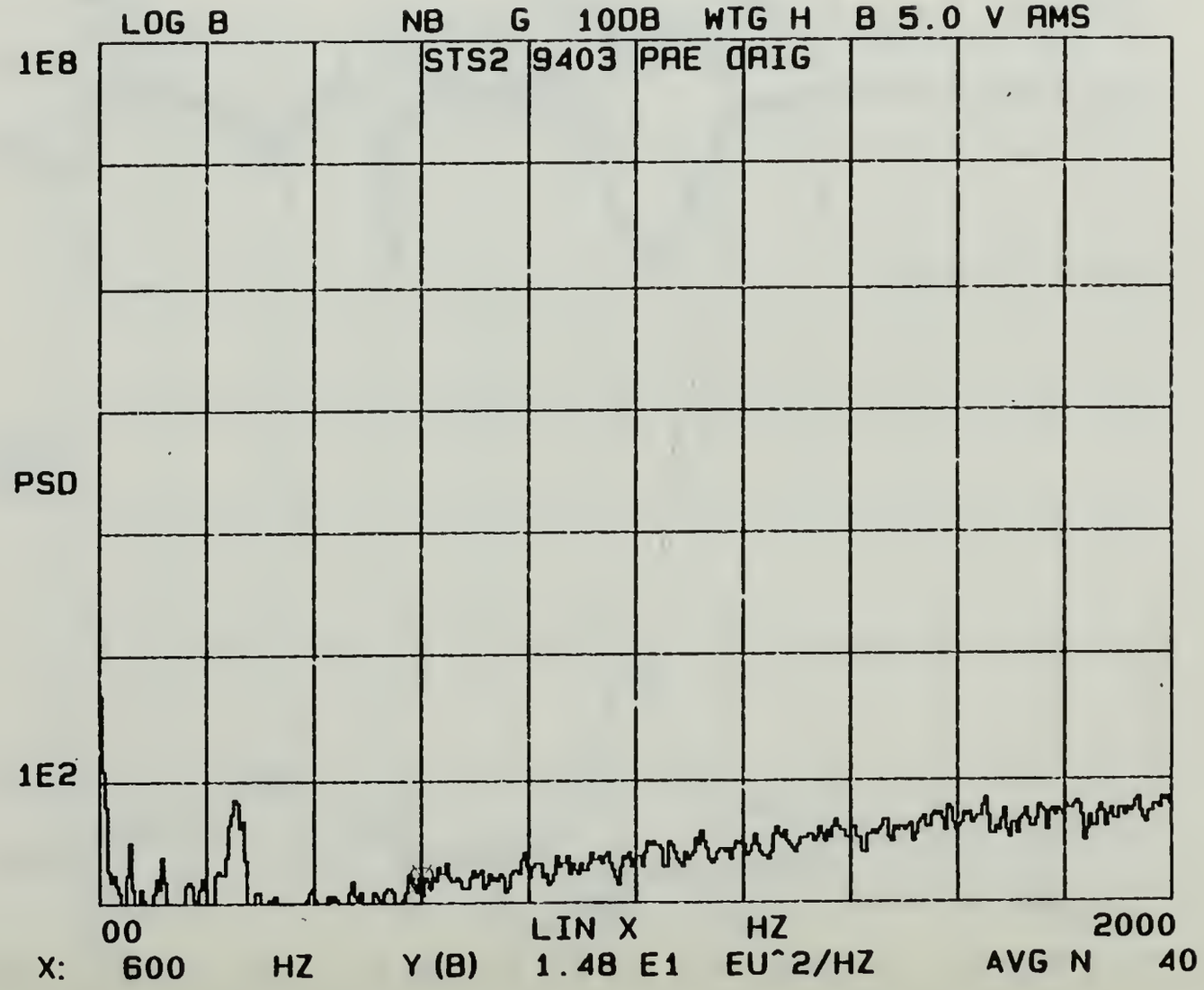
Scale: Linear Ordinate (0-2000 kHz)
Logarithmic Abcissa (10^1 to 10^8 EU)

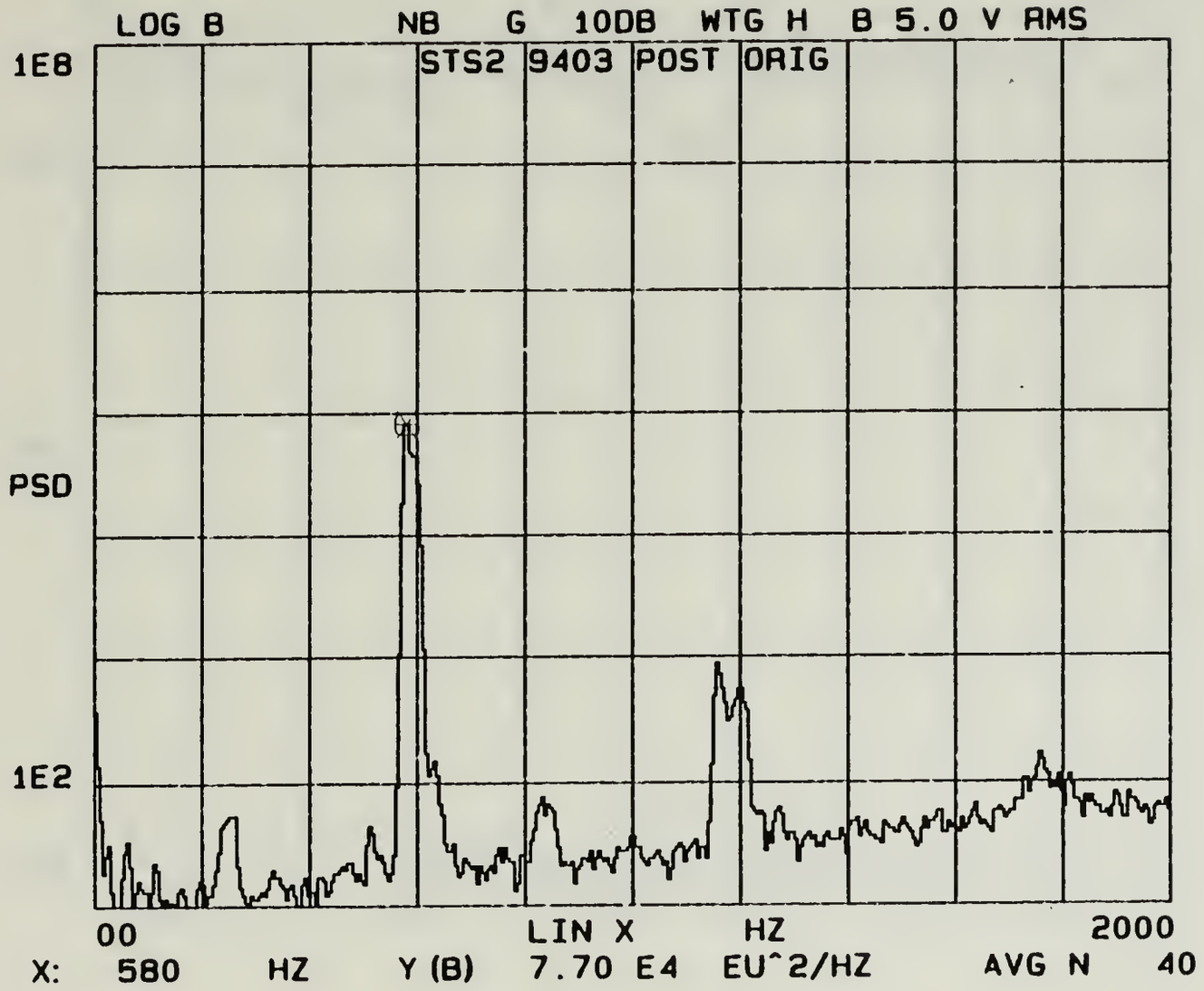

LOG B
NB G 10DB WTG H B 5.0 V RMS
STS2 9405 PRE ORIG
1E8
PSD
1E2
00
LIN X HZ
2000
X: 590 HZ
Y (B) 3.29 E1 EU^2/HZ
AVG N 40

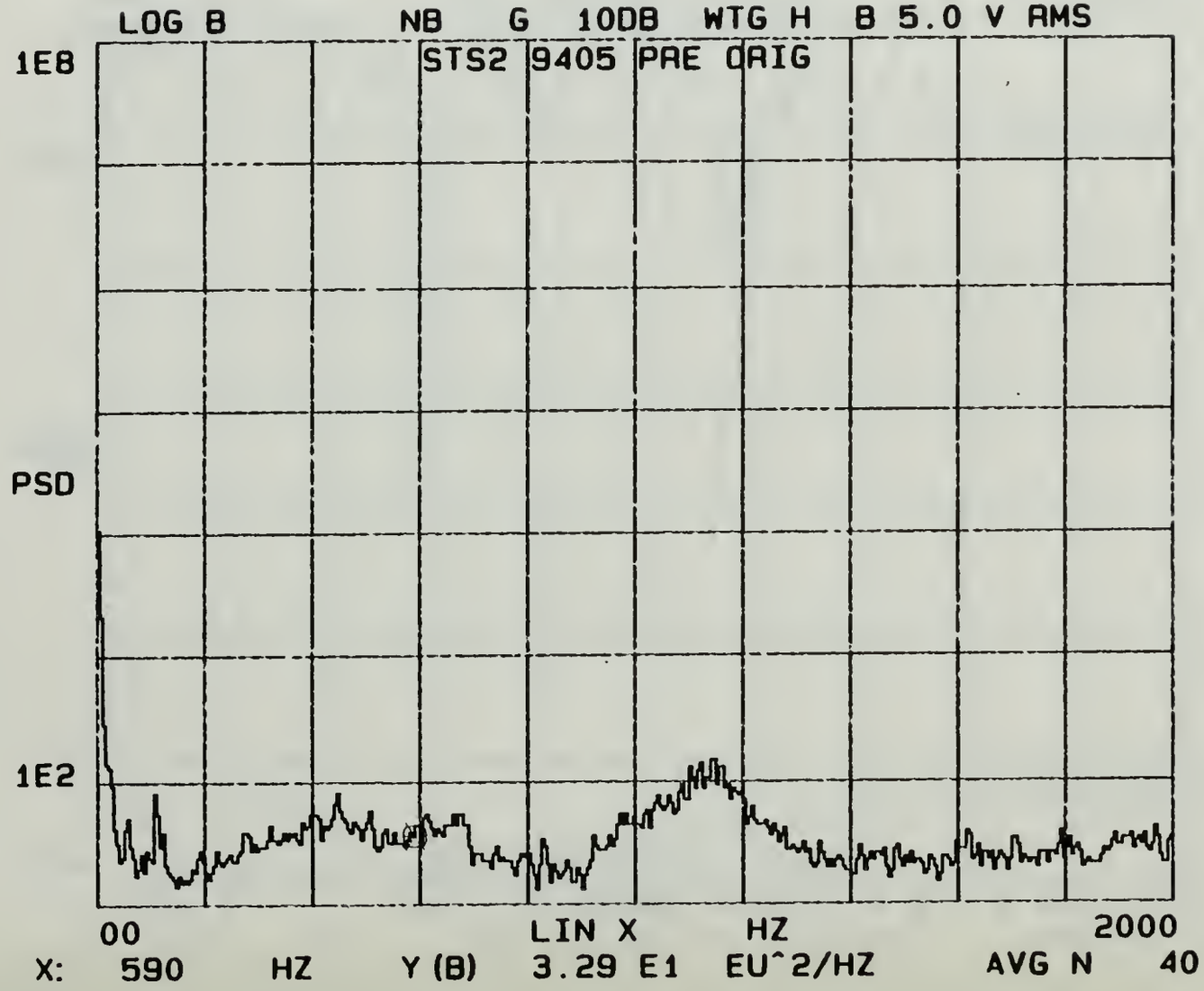

LOG B
NB G 10DB WTG H B 5.0 V RMS
STS2 9405 POST ORIG
1E8
PSD
1E2
00
LIN X
HZ
2000
X: 600 HZ
Y (B) 2.34 E2 EU^2/HZ
AVG N 40

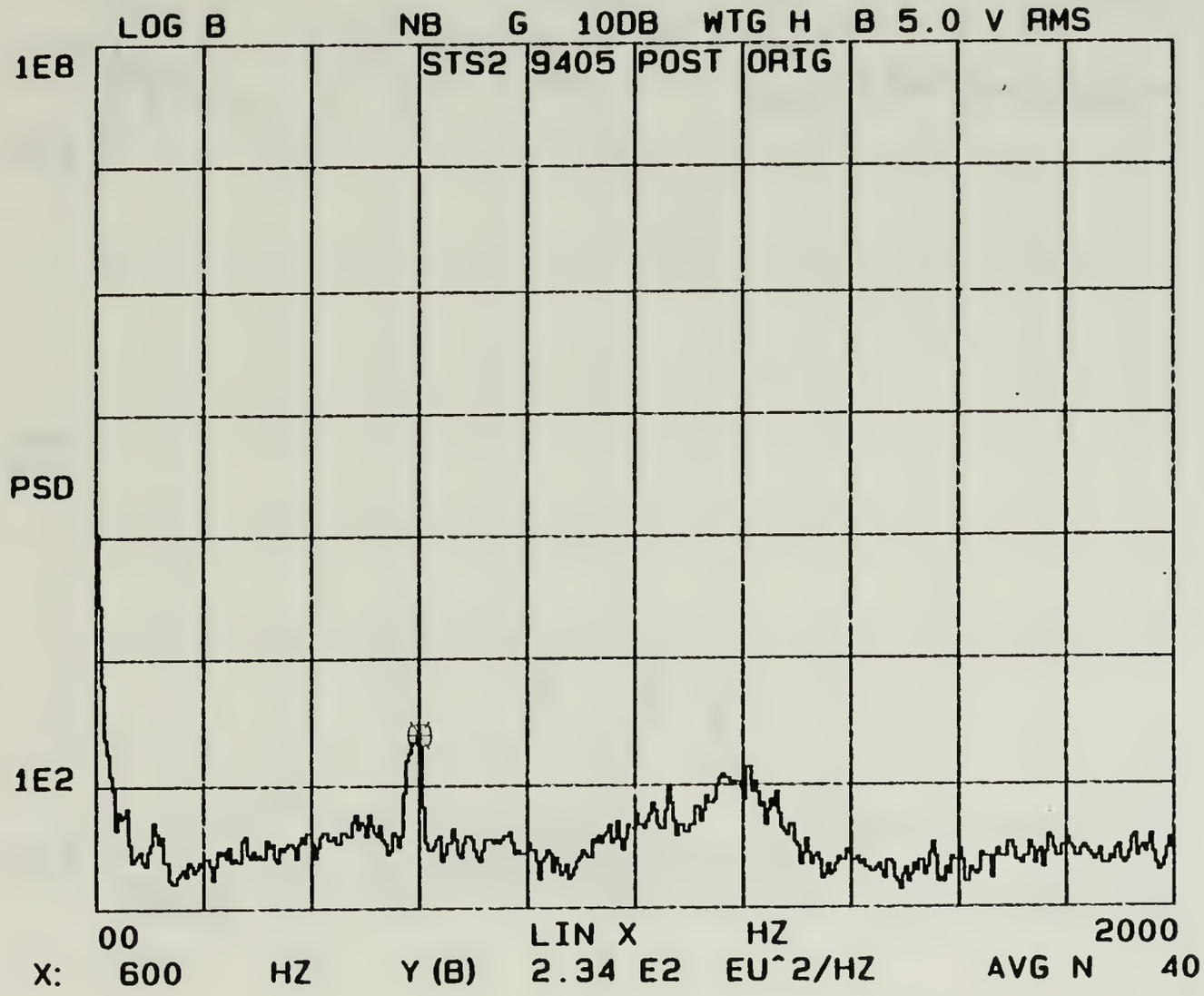

LOG B
NB G 100B WTG H B 5.0 V RMS
STS2 9219 PRE ORIG
1E8
PSD
1E2
00
LIN X HZ
2000
X: 600 HZ Y (B) 4.24 E1 EU^2/HZ AVG N 40

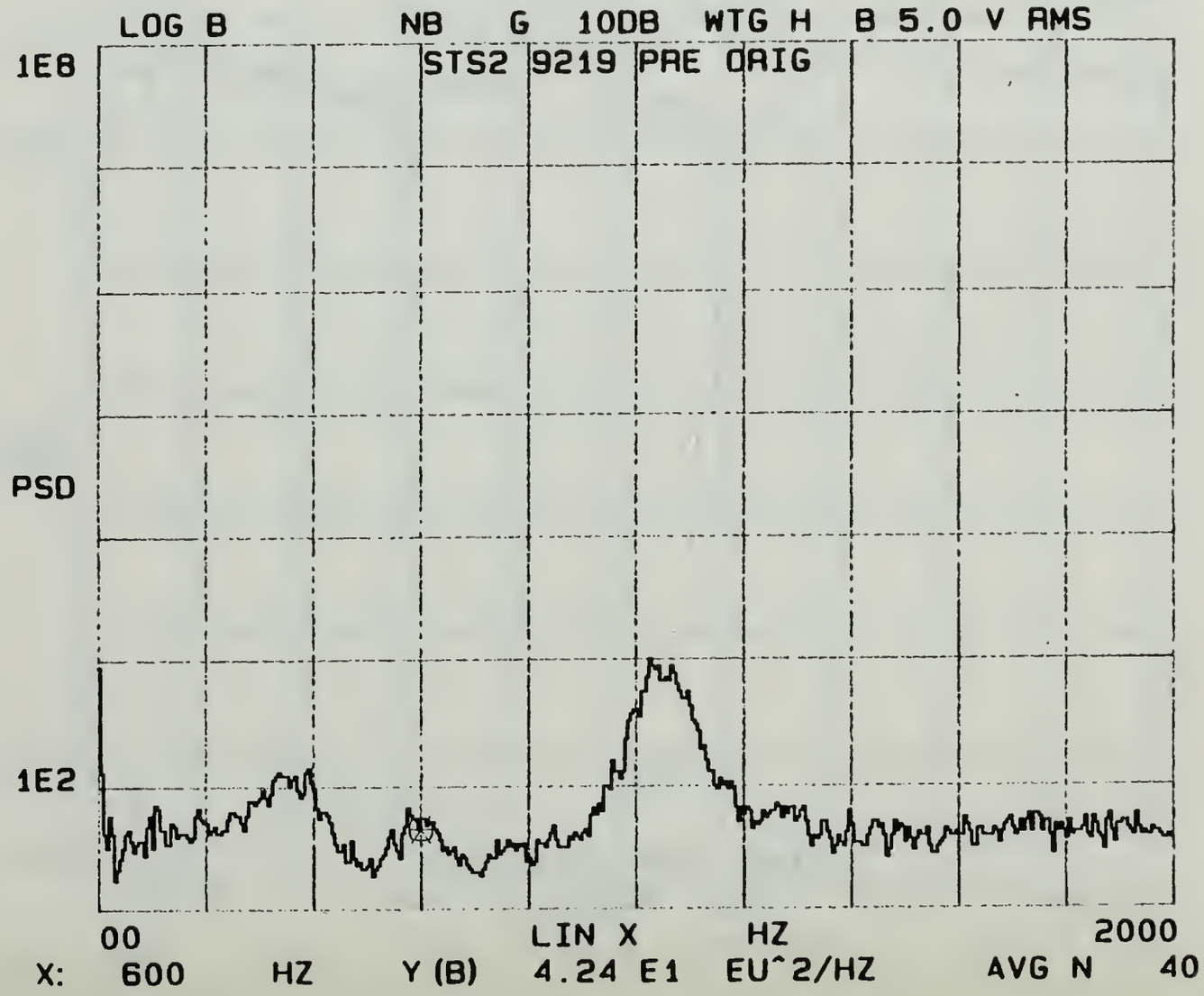

LOG B
NB G 10DB WTG H B 5.0 V RMS
STS2 9219 POST ORIG
1E8
PSD
1E2
00
LIN X HZ
2000
X: 590 HZ
Y (B) 1.11 E3 EU^2/HZ
AVG N 40

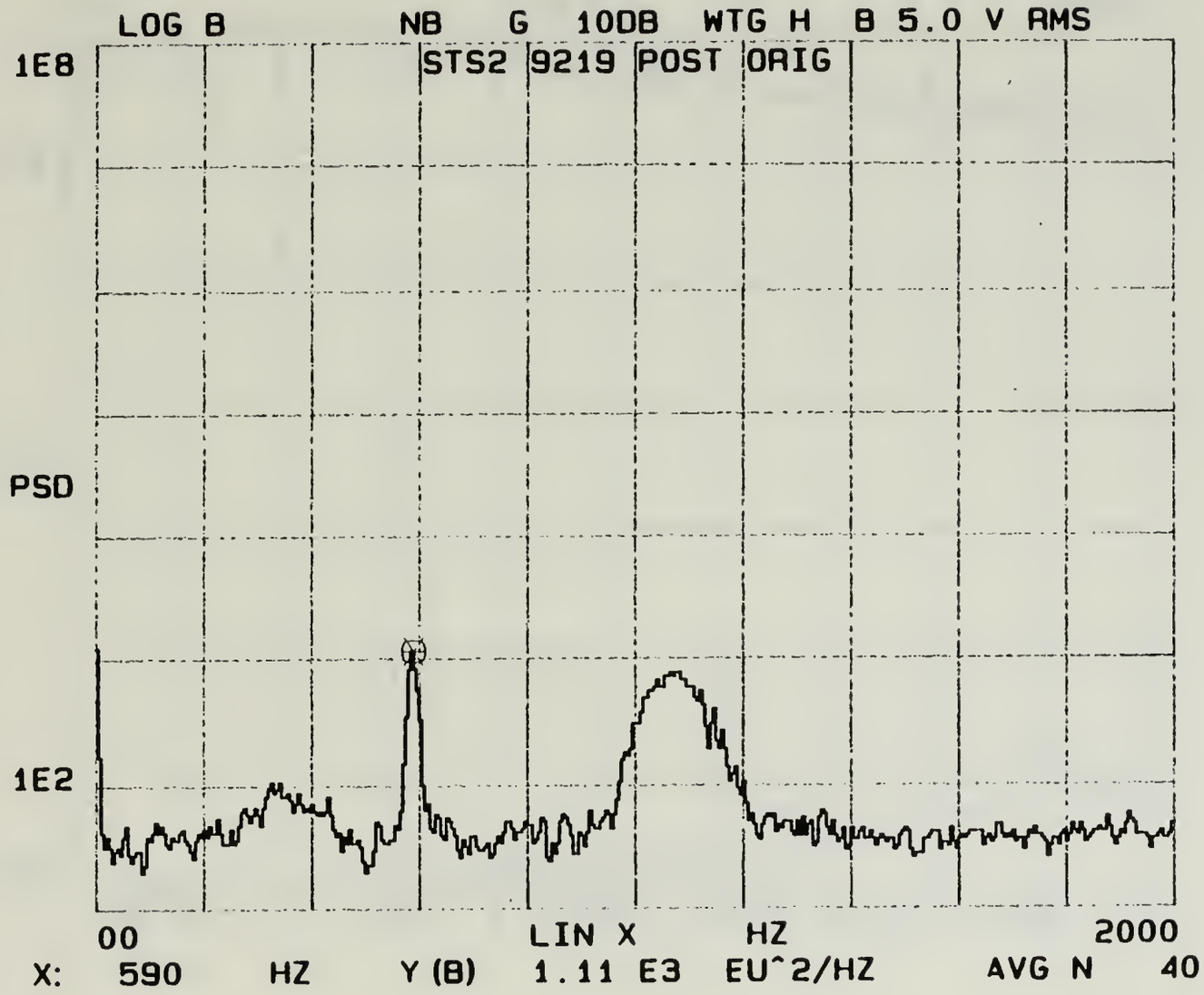

LOG B
NB G 100B WTG H B 5.0 V RMS
STS2 9403 PRE ORIG
1E8
PSD
1E2
00
LIN X HZ
2000
X: 600 HZ
Y (B) 1.48 E1 EU^2/HZ
AVG N 40

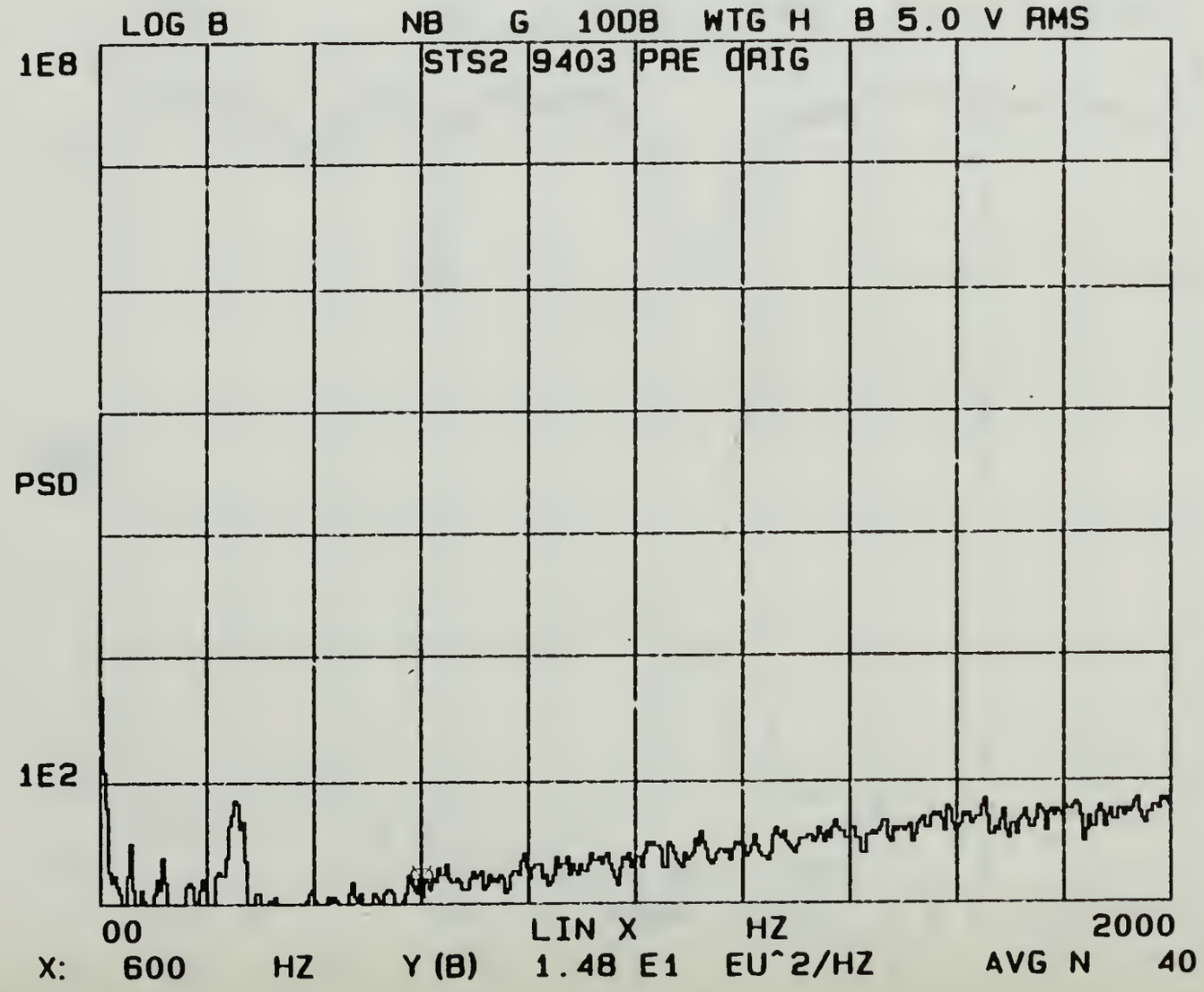

LOG B
NB G 10DB WTG H B 5.0 V RMS
1E8
STS2 9403 POST ORIG
PSD
1E2
00
LIN X
HZ
2000
X: 580 HZ
Y (B) 7.70 E4 EU^2/HZ
AVG N 40

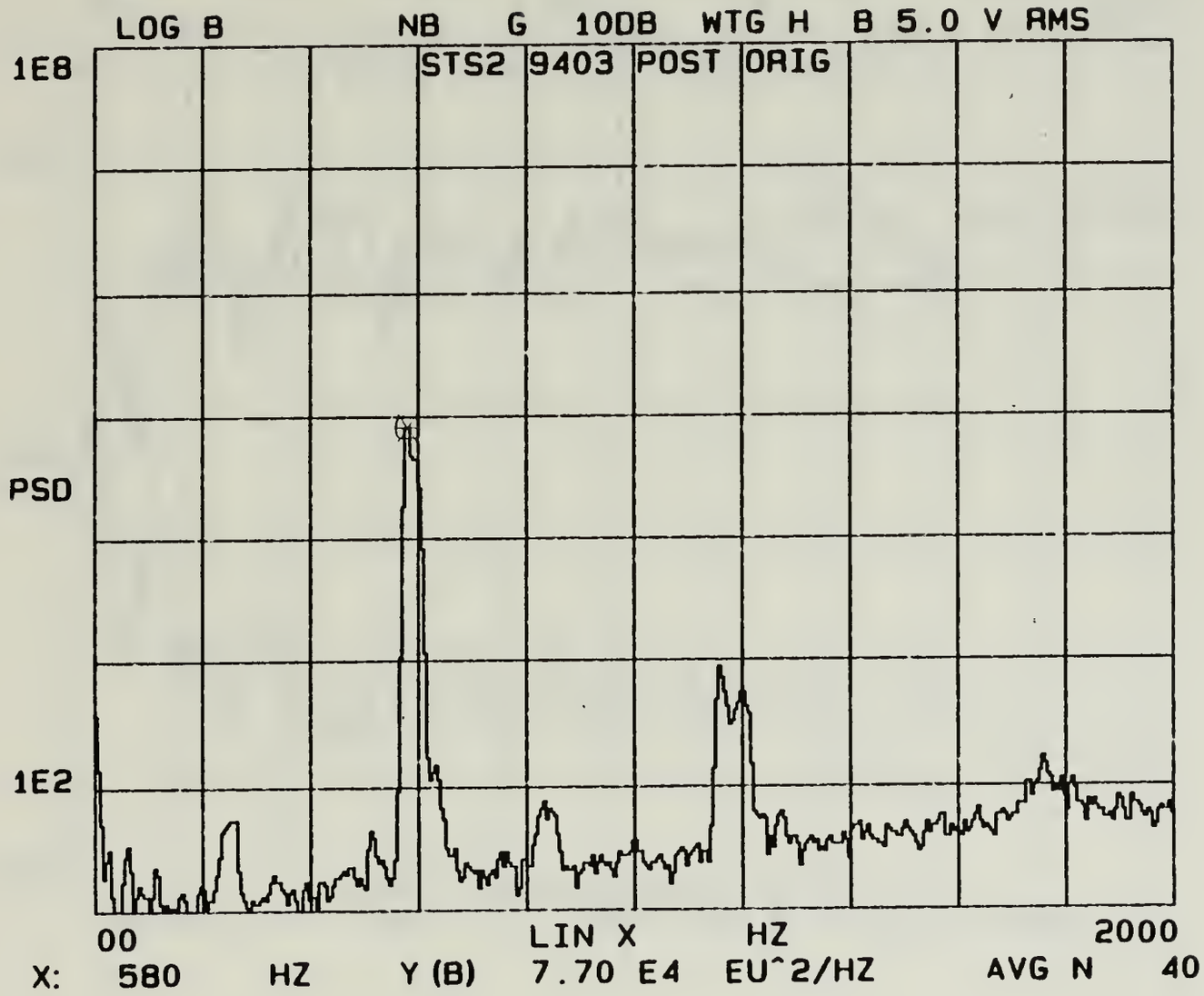

LOG B
NB G 10DB WTG H B 5.0 V AMS
STS2 9405 PRE ORIG
1E8
PSD
1E2
00
LIN X HZ
2000
X: 590 HZ
Y (B) 3.29 E1 EU^2/HZ
AVG N 40

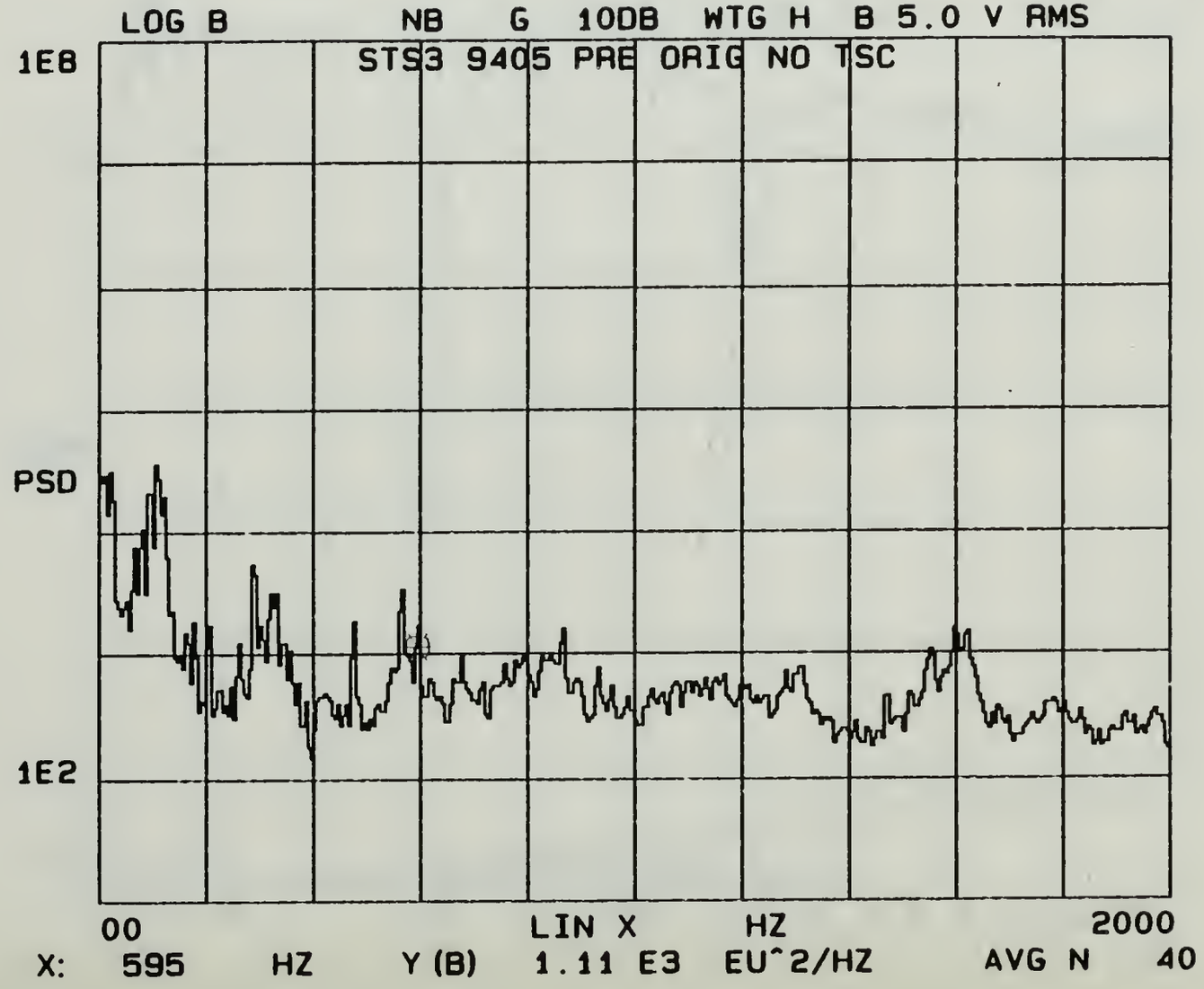

LOG B
NB G 10DB WTG H B 5.0 V RMS
STS2 9405 POST ORIG
1E8
PSD
1E2
00
LIN X HZ
2000
X: 600 HZ Y (B) 2.34 E2 EU^2/HZ AVG N 40

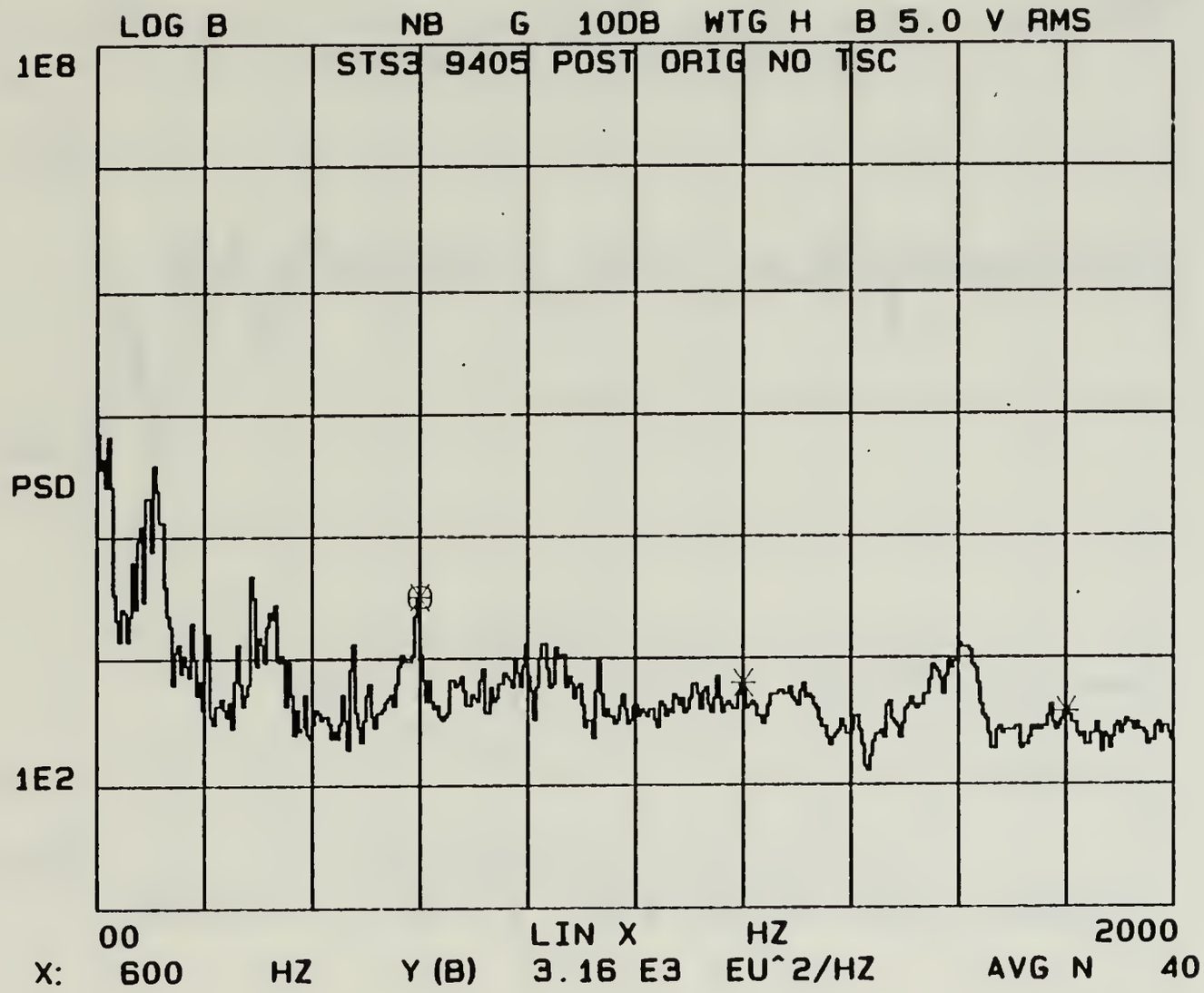

LOG B
NB G 10DB WTG H B 5.0 V RMS
STS2 9219 PRE ORIG
1E8
PSD
1E2
00
LIN X HZ
2000
X: 600 HZ
Y (B) 4.24 E1 EU^2/HZ
AVG N 40

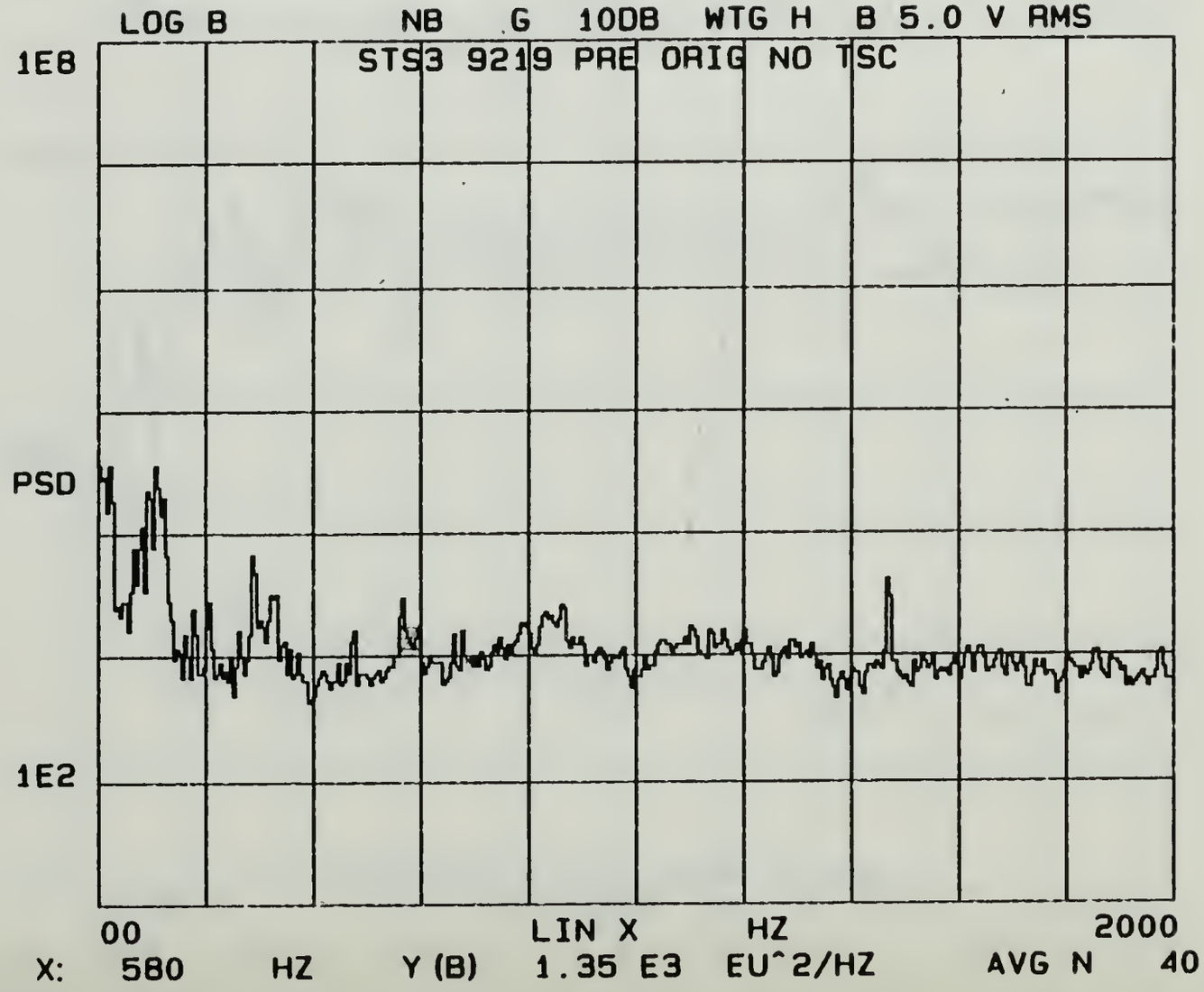

LOG B
NB G 10DB WTG H B 5.0 V RMS
STS2 9219 POST ORIG
1E8
PSD
1E2
00
LIN X HZ
2000
X: 590 HZ Y (B) 1.11 E3 EU^2/HZ AVG N 40

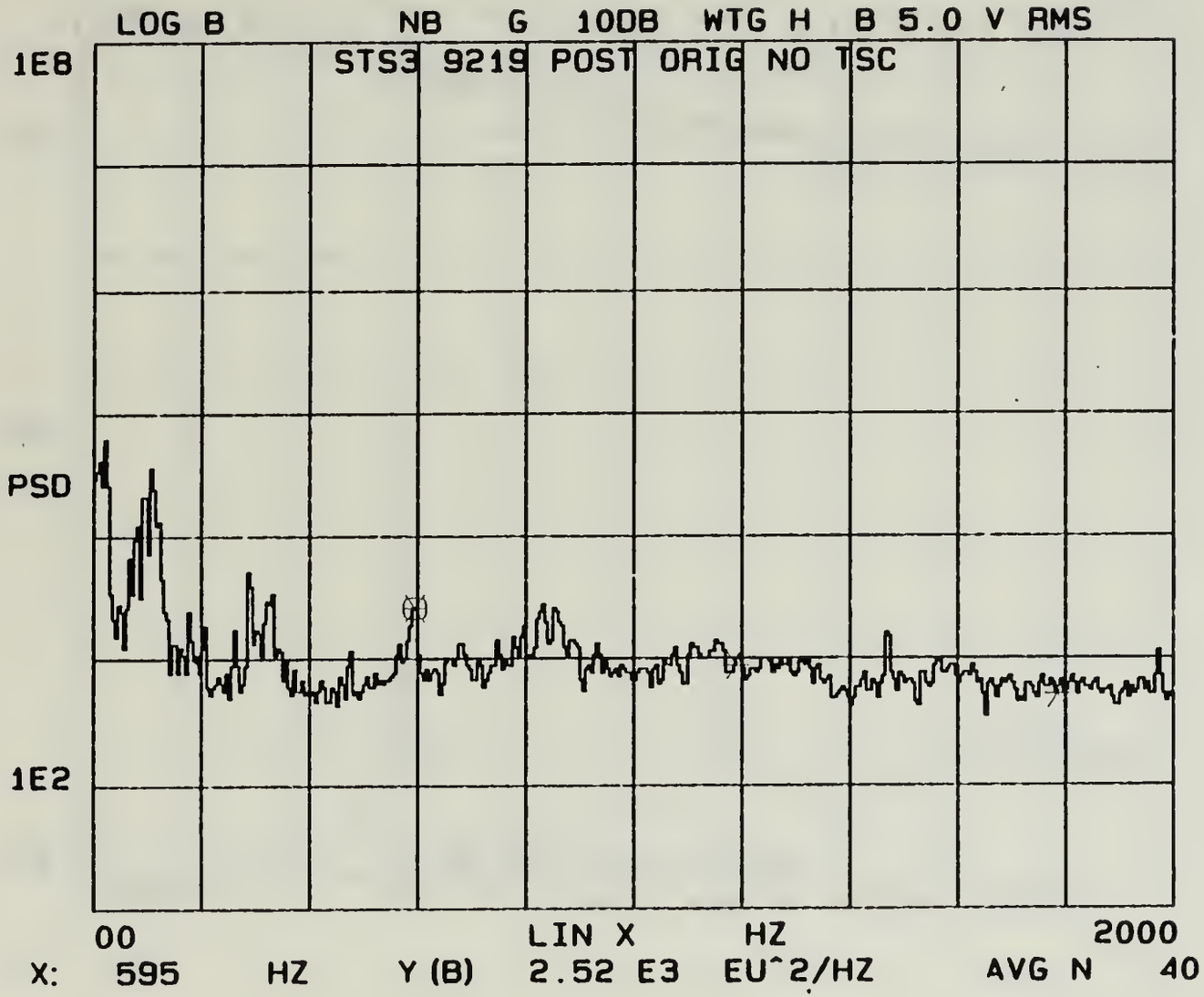

LOG B
NB G 10DB WTG H B 5.0 V RMS
STS2 9403 PRE ORIG
1E8
PSD
1E2
00
LIN X HZ
2000
X: 600 HZ Y (B) 1.48 E1 EU^2/HZ AVG N 40

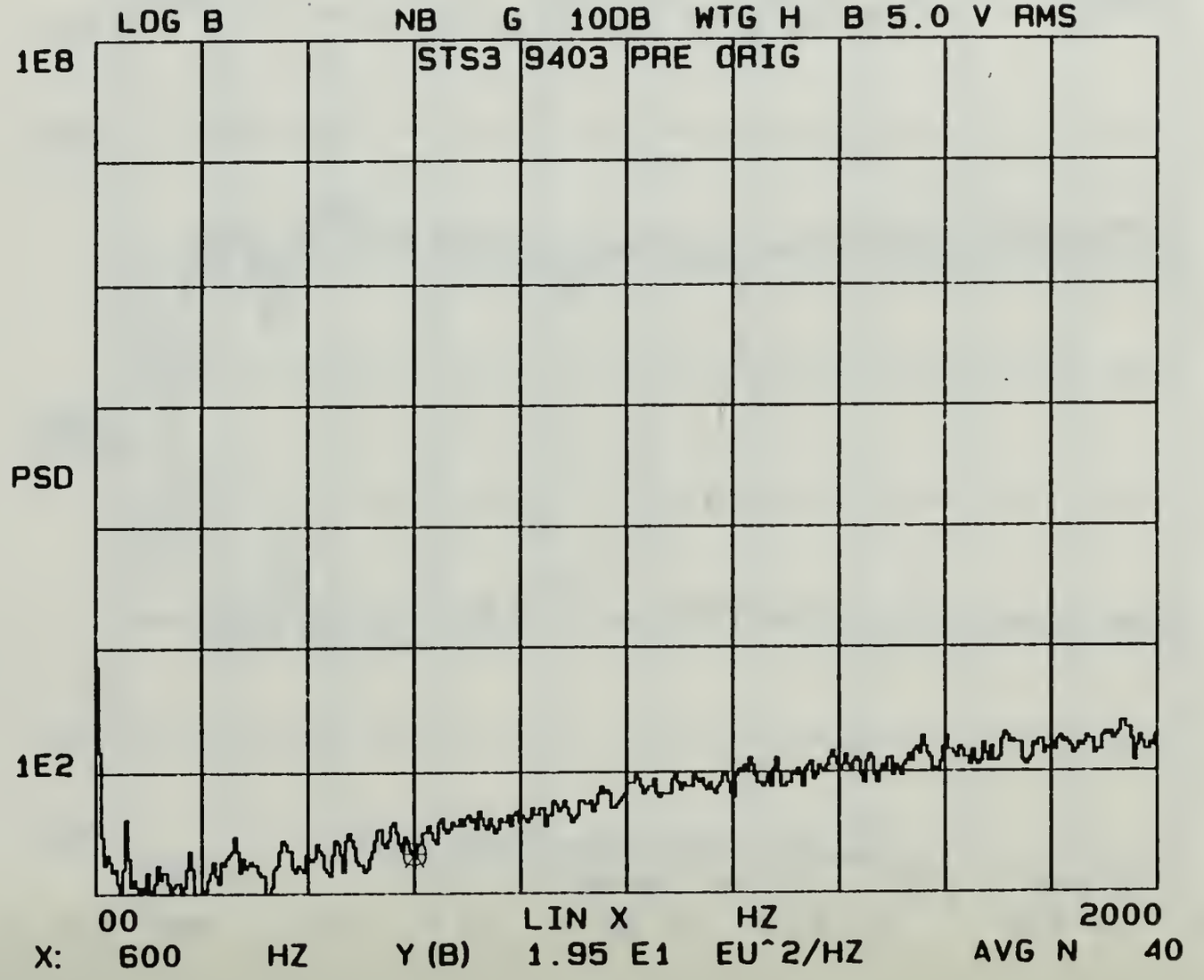

LOG B
NB G 10DB WTG H B 5.0 V RMS
STS2 9403 POST ORIG
1E8
PSD
1E2
00
LIN X HZ
2000
X: 580 HZ
Y (B) 7.70 E4 EU^2/HZ
AVG N 40

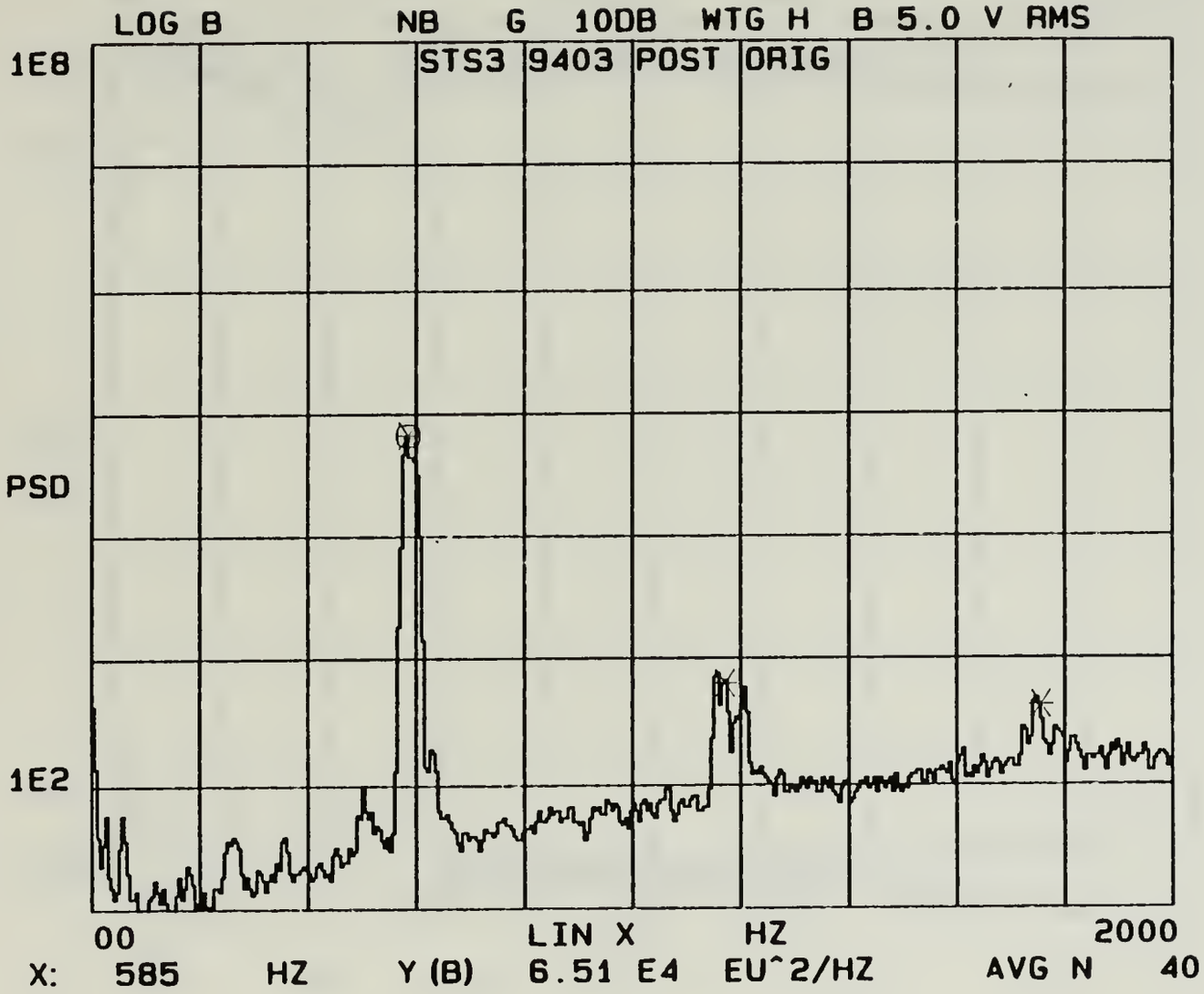

LOG B
NB G 10DB WTG H B 5.0 V RMS
STS3 9405 PRE ORIG NO TSC
1E8
PSD
1E2
00
LIN X HZ
2000
X: 595 HZ Y (B) 1.11 E3 EU^2/HZ AVG N 40

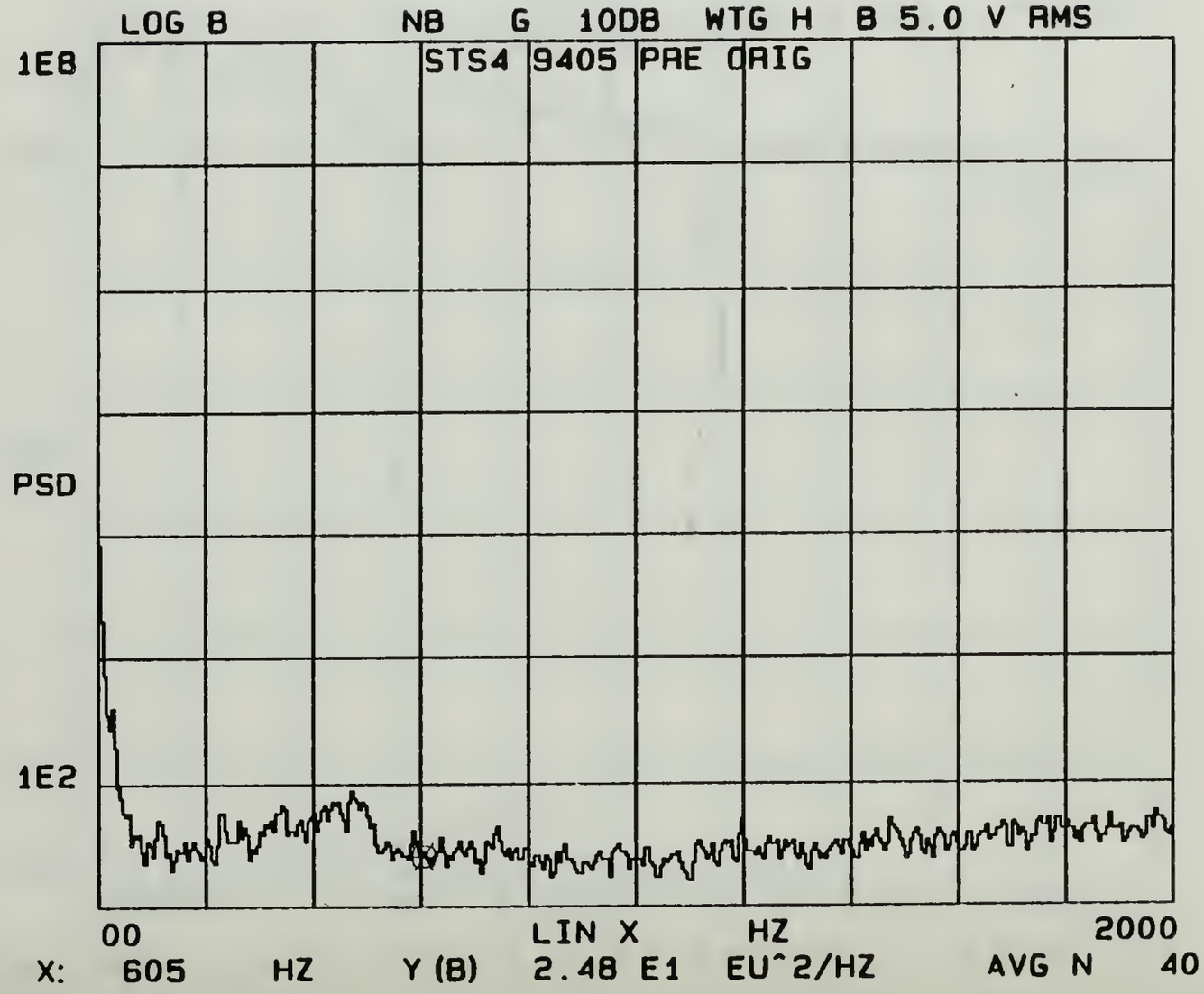

LOG B
NB G 10DB WTG H B 5.0 V RMS
STS3 9405 POST ORIG NO TSC
1E8
PSD
1E2
00
LIN X
HZ
2000
X: 600 HZ
Y (B) 3.16 E3 EU^2/HZ
AVG N 40

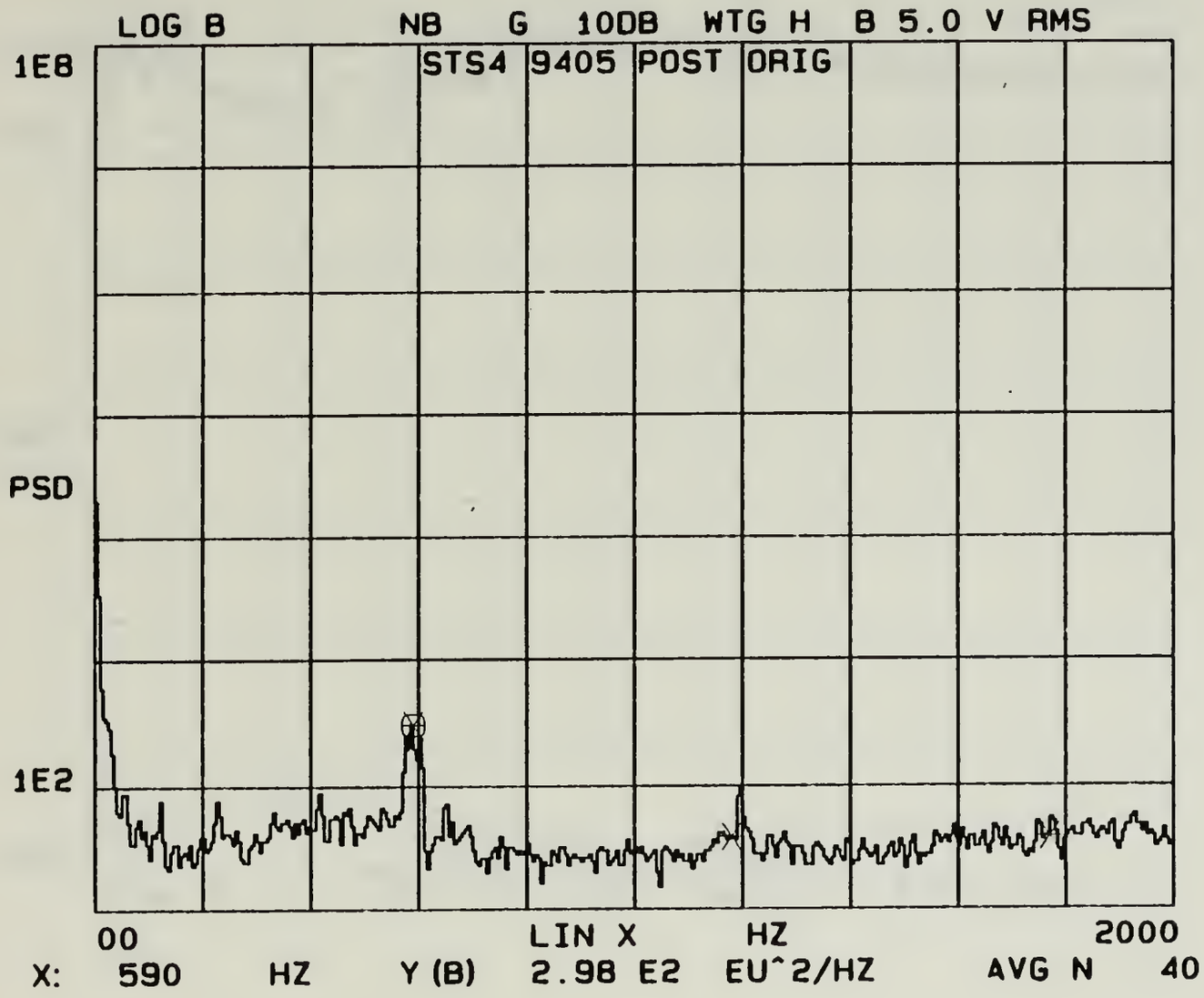

LOG B
NB G 10DB WTG H B 5.0 V RMS
STS3 9219 PRE ORIG NO TSC
1E8
PSD
1E2
00
LIN X HZ
2000
X: 580 HZ
Y (B) 1.35 E3 EU^2/HZ
AVG N 40

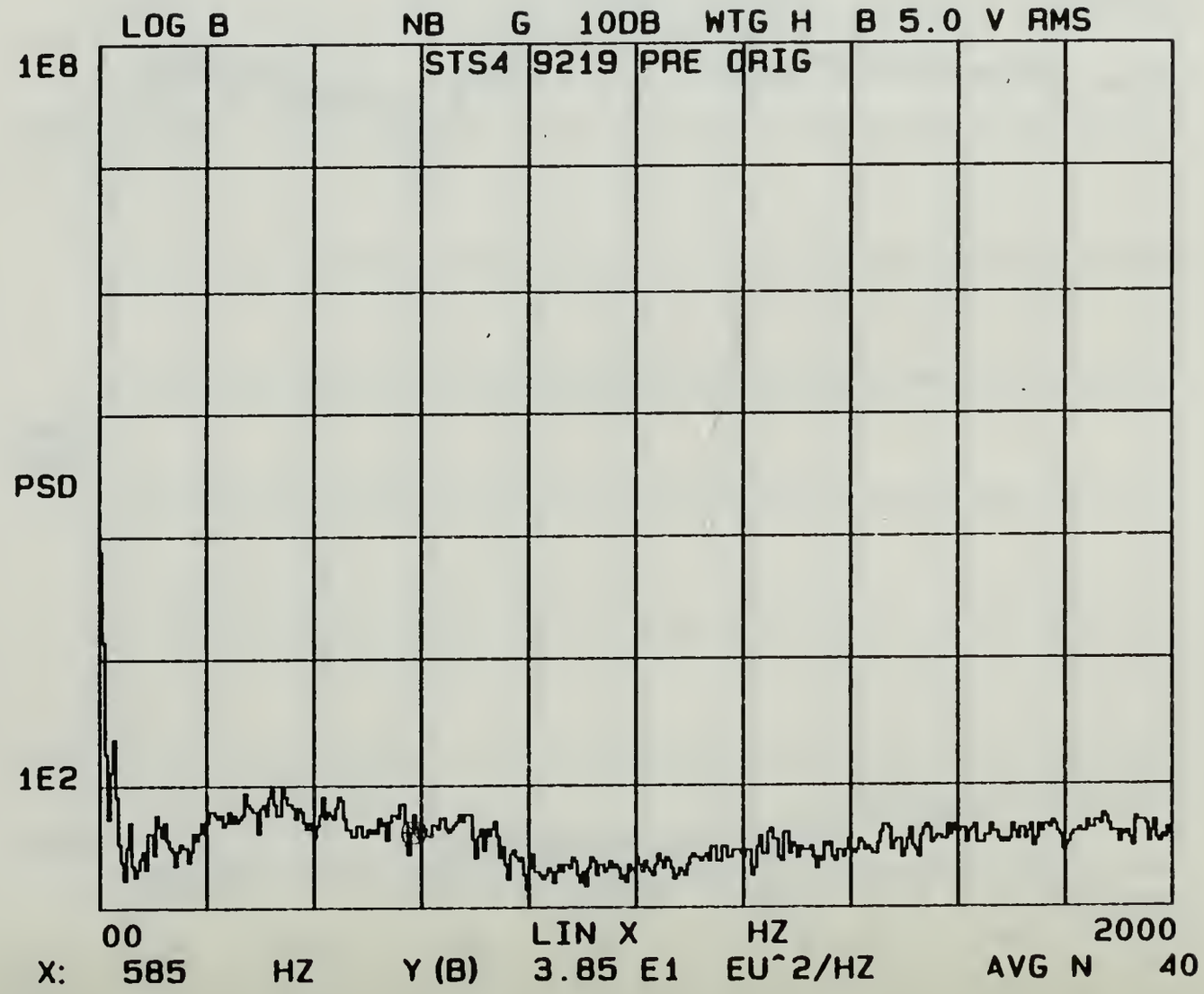

LOG B
NB G 10DB WTG H B 5.0 V RMS
STS3 9219 POST ORIG NO TSC
1E8
PSD
1E2
00
LIN X HZ
2000
X: 595 HZ Y (B) 2.52 E3 EU^2/HZ AVG N 40

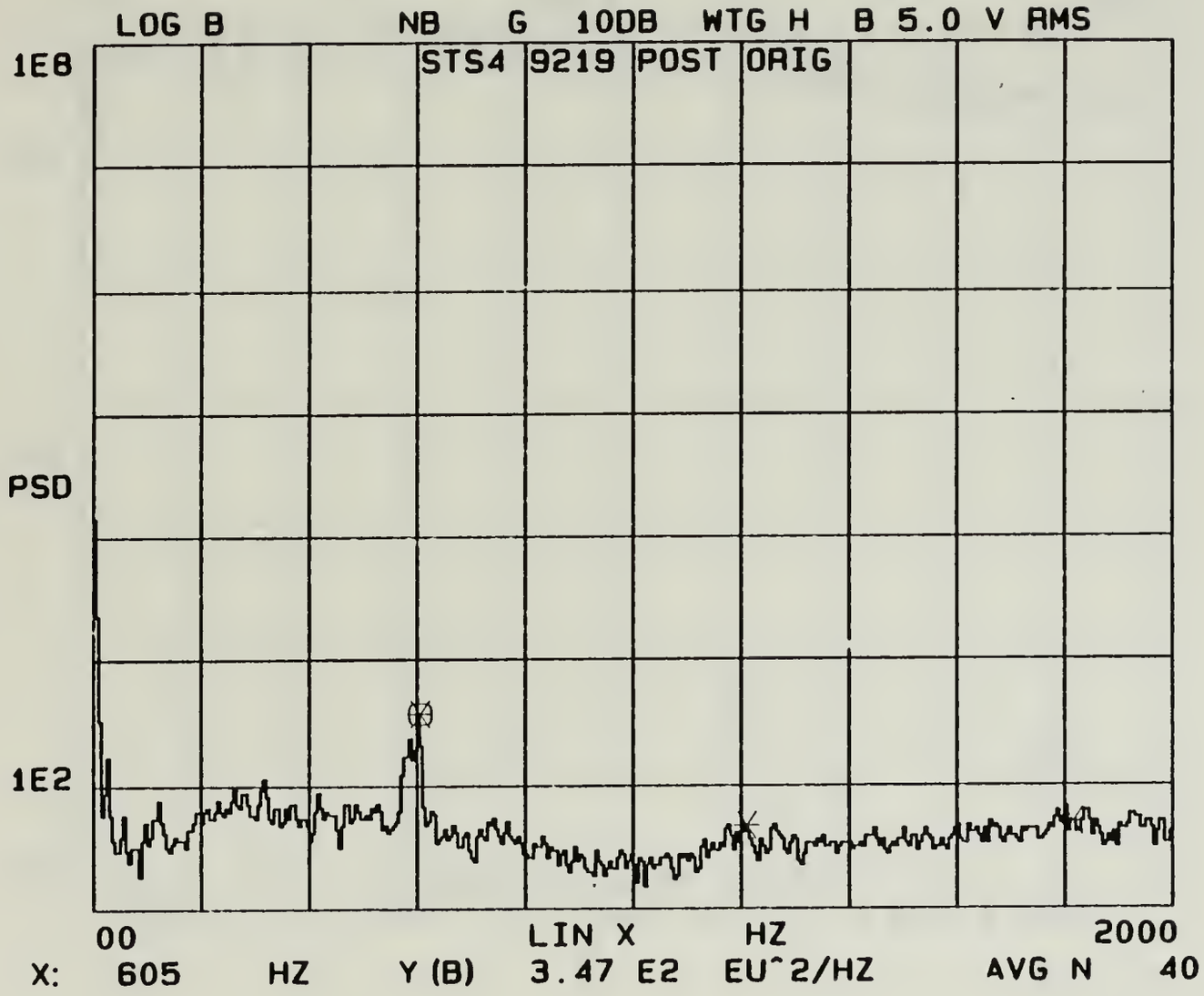

LOG B
NB G 10DB WTG H B 5.0 V RMS
STS3 9403 PRE ORIG
1E8
PSD
1E2
00
LIN X
HZ
2000
X: 600 HZ
Y (B) 1.95 E1 EU^2/HZ
AVG N 40

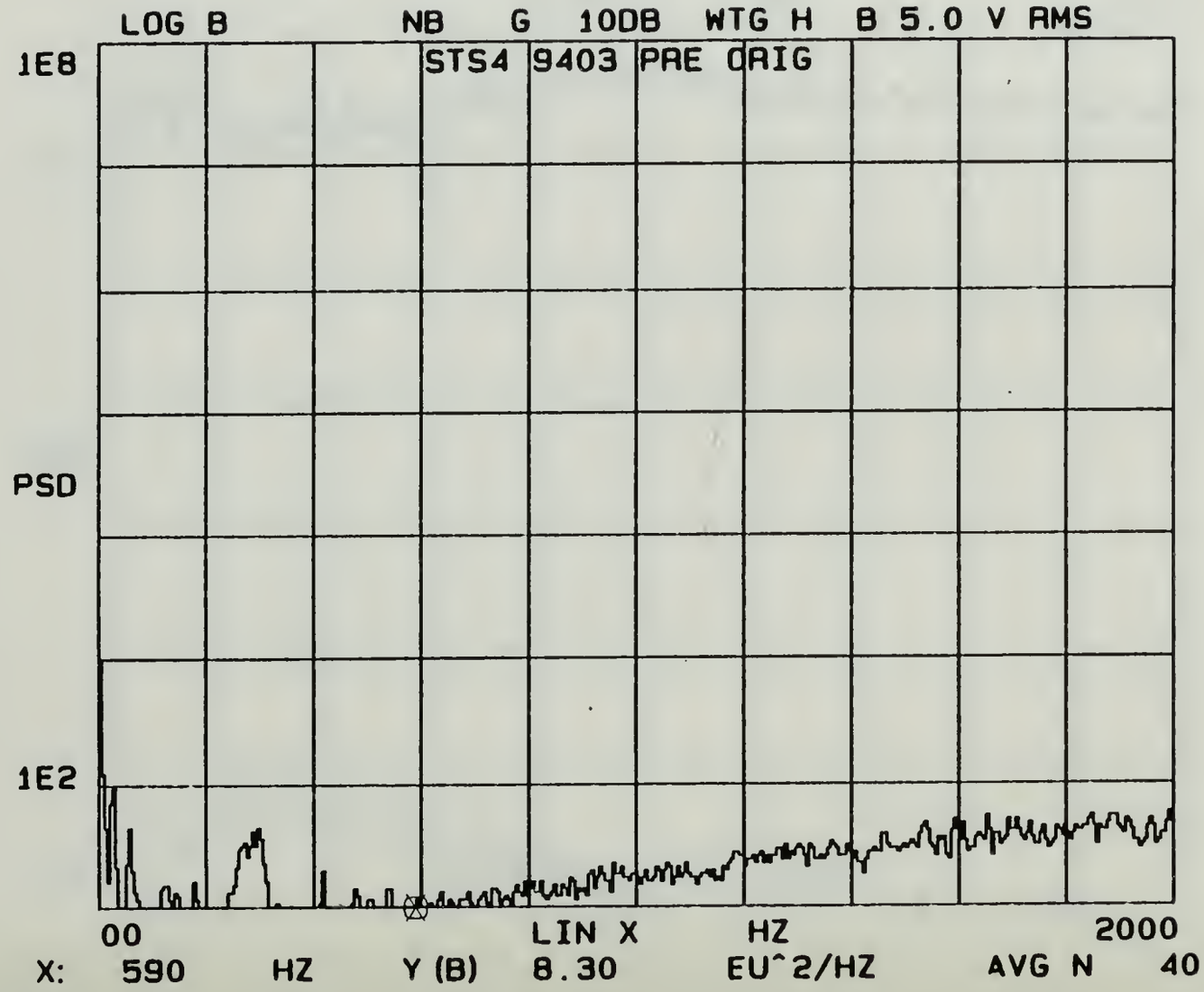

LOG B
NB G 10DB WTG H B 5.0 V RMS
STS3 9403 POST ORIG
1E8
PSD
1E2
00
LIN X HZ
2000
X: 585 HZ
Y (B) 6.51 E4 EU^2/HZ
AVG N 40

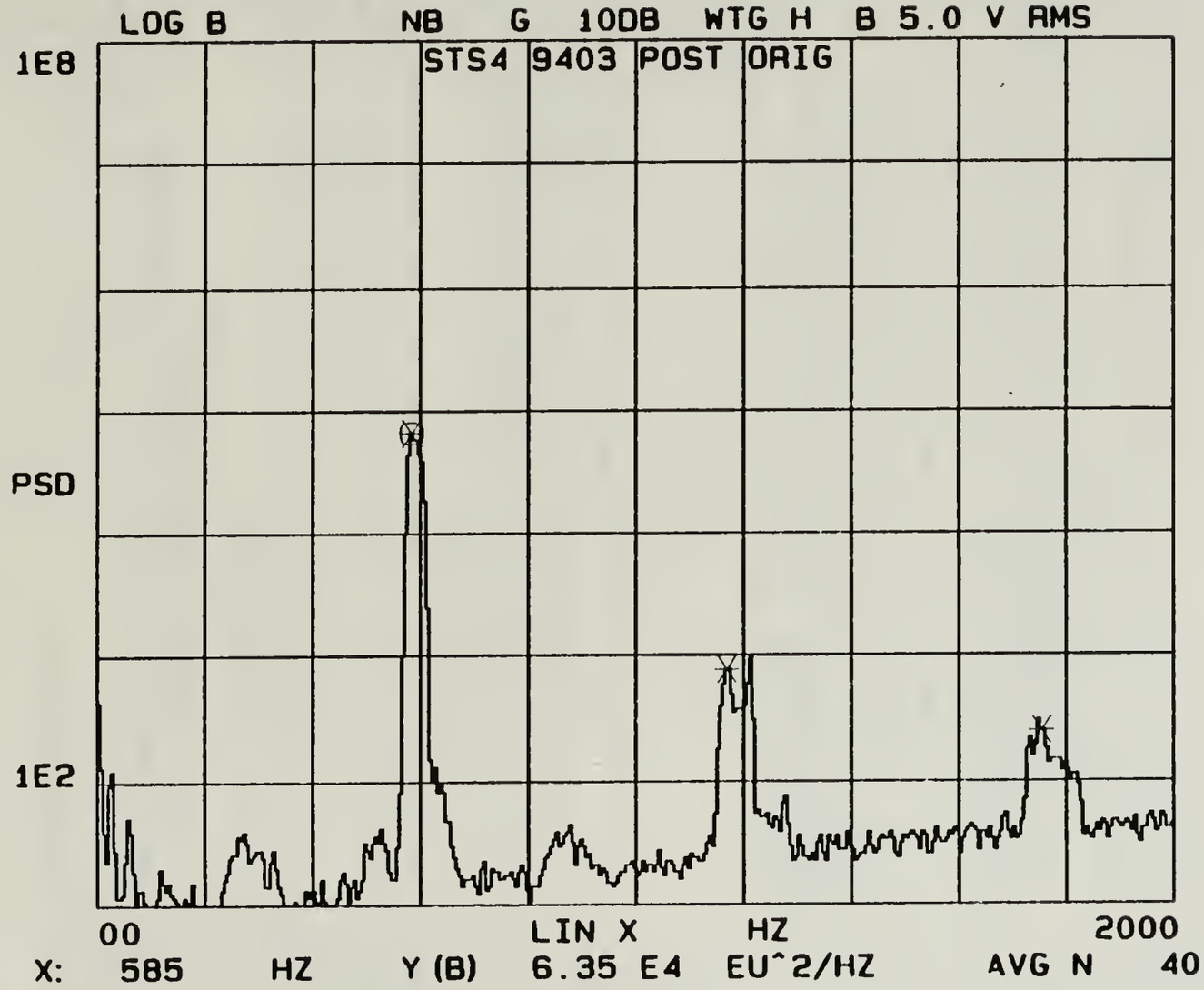

LOG B
NB G 10DB WTG H B 5.0 V RMS
STS4 9405 PRE ORIG
1E8
PSD
1E2
00
LIN X
HZ
2000
X: 605 HZ
Y (B) 2.48 E1 EU^2/HZ
AVG N 40


LOG B
NB G 10DB WTG H B 5.0 V RMS
STS4 9405 POST ORIG
1E8
PSD
1E2
00
LIN X HZ
2000
X: 590 HZ Y (B) 2.98 E2 EU^2/HZ AVG N 40


LOG B
NB G 10DB WTG H B 5.0 V RMS
STS4 9219 PRE ORIG
1E8
PSD
1E2
00
LIN X HZ
2000
X: 585 HZ
Y (B) 3.85 E1 EU^2/HZ
AVG N 40


LOG B
NB G 10DB WTG H B 5.0 V RMS
STS4 9219 POST ORIG
1E8
PSD
1E2
00
LIN X HZ
2000
X: 605 HZ Y (B) 3.47 E2 EU^2/HZ AVG N 40


LOG B
NB G 10DB WTG H B 5.0 V RMS
STS4 9403 PRE ORIG
1E8
PSD
1E2
00
LIN X
HZ
2000
X: 590 HZ
Y (B) 8.30
EU^2/HZ
AVG N 40


LOG B
NB G 10DB WTG H B 5.0 V RMS
STS4 9403 POST ORIG
1E8
PSD
1E2
00
LIN X HZ
2000
X: 585 HZ Y (B) 6.35 E4 EU^2/HZ AVG N 40

FILE: ABPDBP FORTRAN A1

```
C*************************************************************************ABP00010
C                                                                         *ABP00020
C                              APPENDIX C                                 *ABP00030
C                                                                         *ABP00040
C              FORTRAN PROGRAM ABPDBP AND PROGRAM OUTPUT                  *ABP00050
C                                                                         *ABP00060
C                                                                         *ABP00070
C     THIS PROGRAM CALCULATES:                                            *ABP00080
C                                                                         *ABP00090
C     1)  ANALOG ELLIPTIC BANDPASS FILTER RESISTOR AND CAPACITOR          *ABP00100
C         VALUES (STARTING FROM TABULATED PARAMETERS CORRESPONDING        *ABP00110
C         TO THE DESIRED FILTER RESPONSE)                                 *ABP00120
C                                                                         *ABP00130
C     2)  ANALOG ELLIPTIC BANDPASS TRANSFER FUNCTION (IF DESIRED FOR      *ABP00140
C         DIGITAL TRANSFORMATION THEN MUST MODIFY CODE IN THIS PROGRAM   *ABP00150
C         SECTION AND SUBSTITUTE WODIG FOR W0 TO REALIZE NECESSARY        *ABP00160
C         PRE-WARPING COMPENSATION)                                       *ABP00170
C                                                                         *ABP00180
C         A)  ANALOG TRANSFER FUNCTION COMPLEX ZEROS AND POLES            *ABP00190
C                                                                         *ABP00200
C     3)  DIGITAL TRANSFER FUNCTION (BY APPLICATION OF THE BILINEAR       *ABP00210
C         TRANSFORM TO THE PRE-WARPED CASE OF THE ANALOG TRANSFER         *ABP00220
C         FUNCTION, WHICH IS PROVIDED IN SECTION 2 BY USING THE           *ABP00230
C         PRE-WARPED FREQUENCY ANALOG -- WODIG (VICE W0) -- IN THE        *ABP00240
C         ANALOG TRANSFER FUNCTION COMPUTATION).  THE BILINEAR            *ABP00250
C         TRANSFORM IS ACCOMPLISHED BY THE FOLLOWING SUBSTITUTION:        *ABP00260
C                                                                         *ABP00270
C                                    Z - 1                                *ABP00280
C                          S = (2/T) -------                              *ABP00290
C                                    Z + 1                                *ABP00300
C                                                                         *ABP00310
C         WHERE T IS THE SAMPLING FREQUENCY OF THE DIGITAL SYSTEM.        *ABP00320
C                                                                         *ABP00330
C         A)  DIGITAL TRANSFER FUNCTION COMPLEX ZEROS AND POLES           *ABP00340
C                                                                         *ABP00350
C     4)  POLYNOMIAL COEFFICIENTS FOR FIRST AND SECOND ORDER CASCADED     *ABP00360
C         TERMS WHICH WILL BE USED TO PERFORM A 2920 ANALOG/DIGITAL       *ABP00370
C         SIGNAL PROCESSING SIMULATION                                    *ABP00380
C                                                                         *ABP00390
C*************************************************************************ABP00400
C     TYPE DECLARATIONS                                                    ABP00410
C*************************************************************************ABP00420
C SECTION 1                                                               ABP00430
      REAL A,B,C,D,E,A1,Q,K,K1,K2,W0,PI,F0,WODIG,FODIG,T,TDIV2            ABP00440
      REAL C11,C12,C21,C22                                                ABP00450
      REAL R11,R12,R13,R14,R15,R16,R17                                    ABP00460
      REAL R21,R22,R23,R24,R25,R26,R27                                    ABP00470
C SECTION 2                                                               ABP00480
      INTEGER INUM1,INUM2,IDEN1,IDEN2,IANUM,IADEN                         ABP00490
      REAL RHO,RHO1,RHO2,ALFA1,ALFA2,BETA1,BETA2,GAMMA1,GAMMA2            ABP00500
      REAL NUM1(3),NUM2(3),DEN1(3),DEN2(3)                                ABP00510
      REAL ANUM(5),ADEN(5)                                                ABP00520
C SECTION 2A                                                              ABP00530
      INTEGER IERR,IPDEG                                                  ABP00540
      REAL ANMINV(5),ADNINV(5)                                            ABP00550
```

FILE: ABPDBP FORTRAN A1

```
      COMPLEX AZERO(4),APOLE(4)                                          ABP00560
C SECTION 3                                                              ABP00570
      INTEGER IZM,IZP,IDNUM,IDDEN,IDTMP                                  ABP00580
      REAL TNCOEF,TDCOEF                                                 ABP00590
      REAL ZM1(5),ZM2(5),ZM3(5),ZM4(5),ZP1(5),ZP2(5),ZP3(5),ZP4(5)       ABP00600
      REAL DNUM(5),DDEN(5),DTMP(9)                                       ABP00610
C SECTION 3A                                                             ABP00620
      REAL DNMINV(5),DDNINV(5)                                           ABP00630
      REAL RZ(4),RP(4)                                                   ABP00640
      COMPLEX DZERO(4),DPOLE(4)                                          ABP00650
C SECTION 4                                                              ABP00660
      INTEGER I4,I2                                                      ABP00670
      COMPLEX*16 DZ4(4),DP4(4),DZ21(2),DP21(2),DZ22(2),DP22(2)           ABP00680
      COMPLEX*16 DN4(4),DD4(4),DN21(2),DD21(2),DN22(2),DD22(2)           ABP00690
      COMPLEX DN45(5),DD45(5),DN213(3),DD213(3),DN223(3),DD223(3)        ABP00700
C*********************************************************************ABP00710
C    TABULATED INPUT PARAMETERS FOR DESIRED SECOND ORDER LOWPASS         ABP00720
C    FILTER EQUIVALENT HAVING THE FOLLOWING CHARACTERISTICS:             ABP00730
C              N = 2                                                     ABP00740
C              MAXIMUM PASSBAND RIPPLE WIDTH (PRW) = 2.0 DB              ABP00750
C              MINIMUM LOSS IN THE STOPBAND (MSL) = 30.0 DB              ABP00760
C              NORMALIZED TRANSITION WIDTH = 2.2921                      ABP00770
C              FILTER GAIN (K) = 1.0                                     ABP00780
C*********************************************************************ABP00790
      F0=600.                                                            ABP00800
      A=21.164003                                                        ABP00810
      B=0.787152                                                         ABP00820
      C=0.842554                                                         ABP00830
      Q=12.                                                              ABP00840
      PI=3.1415927                                                       ABP00850
      K=1.                                                               ABP00860
C*********************************************************************ABP00870
C    DERIVED PARAMETERS                                                  ABP00880
C*********************************************************************ABP00890
      E=(1./B)*SQRT((C+4.*Q**2.+SQRT((C+4.*Q**2.)**2.-(2.*B*Q)**2.))/2.)ABP00900
      D=.5*((B*E/Q)+SQRT((B*E/Q)**2.-4.))                                ABP00910
      A1=1.+(1./(2.*Q**2.))*(A+SQRT(A**2.+4.*A*Q**2.))                   ABP00920
      K1=SQRT(K)                                                         ABP00930
      K2=K1                                                              ABP00940
      T=4.*192./6.666E+06                                                ABP00950
      TDIV2=T/2.                                                         ABP00960
      W0=2.*PI*F0                                                        ABP00970
      WODIG=(1./TDIV2)*ATAN(W0*TDIV2)                                    ABP00980
      FODIG=WODIG/(2.*PI)                                                ABP00990
      WRITE(4,10)A,B,C,D,E,A1,F0,W0,Q,K,K1,K2,WODIG,FODIG                ABP01000
C*********************************************************************ABP01010
C                                                                       *ABP01020
C                            SECTION 1                                  *ABP01030
C                                                                       *ABP01040
C    THIS PROGRAM SECTION COMPUTES ANALOG ELLIPTIC BANDPASS FILTER      *ABP01050
C    RESISTOR AND CAPACITOR VALUES USING THE ABSOLUTE AND DERIVED       *ABP01060
C    PARAMETERS CALCULATED ABOVE:                                       *ABP01070
C                                                                       *ABP01080
C    CHOOSE C11=C21=.01E-06 (APPROX)=.01 UF                             *ABP01090
C    CHOOSE C12=C22=.01E-06(APPROX)=.01 UF                              *ABP01100
```

FILE: ABPDBP FORTRAN A1

```
C     CHOOSE R17=R27=1./(W0*C1)(APPROX)=26.5 KOHMS                      *ABP01110
C     USE MEASURED VALUES FOR CALCULATIONS                              *ABP01120
C                                                                       *ABP01130
C***********************************************************************ABP01140
      C11=.00996E-06                                                     ABP01150
      C12=.00995E-06                                                     ABP01160
      C21=.01030E-06                                                     ABP01170
      C22=.01034E-06                                                     ABP01180
      R17=26.7E+03                                                       ABP01190
      R27=26.7E+03                                                       ABP01200
C***********************************************************************ABP01210
C     ELLIPTIC BANDPASS FILTER STAGE RESISTOR VALUES                     ABP01220
C***********************************************************************ABP01230
      R11=(E/(K*D*W0*C11))*SQRT(A/C)                                     ABP01240
      R12=K*R11*SQRT(C/A)                                                ABP01250
      R13=1./(D*W0*C11)                                                  ABP01260
      R14=(R17/K)*SQRT(A/C)                                              ABP01270
      R15=(D/(K*A1*W0*C12))*SQRT(A/C)                                    ABP01280
      R16=C11*R13/C12                                                    ABP01290
      WRITE(4,20)C11,C12,R17,R11,R12,R13,R14,R15,R16                     ABP01300
      R21=((D*E)/(K*W0*C21))*SQRT(A/C)                                   ABP01310
      R22=(K*R21)*SQRT(C/A)                                              ABP01320
      R23=D/(W0*C21)                                                     ABP01330
      R24=(R27/K)*SQRT(A/C)                                              ABP01340
      R25=(A1/(K*D*W0*C22))*SQRT(A/C)                                    ABP01350
      R26=C21*R23/C22                                                    ABP01360
      WRITE(4,30)C21,C22,R27,R21,R22,R23,R24,R25,R26                     ABP01370
   10 FORMAT(' SECTION 1 OUTPUT',//,                                     ABP01380
     &' INPUT AND DERIVED PARAMETERS FOR FURTHER CALCULATIONS'           ABP01390
     &,//,' A = ',F9.6,/,' B = ',F9.6,/,' C = ',F9.6,/,' D = ',         ABP01400
     &F9.6,/,' E = ',F9.6,/,' A1 = ',F9.6,/,' F0 = ',F9.3,/,' W0 = '    ABP01410
     &,F9.3,/,' Q = ',F9.6,/,' K = ',F9.6,/,' K1 = ',F9.6,/,            ABP01420
     &' K2 ',F9.6,/,' W0(DIG) = ',F9.3,/,' F0(DIG) = ',F9.3,///)        ABP01430
   20 FORMAT(' ELLIPTIC ANALOG BPF COMPONENT VALUES',//,' FIRST STAGE'  ABP01440
     &,//,' C11 = ',E8.3,/,' C12 = ',E8.3,/,' R17 = ',E8.3,             ABP01450
     &/,' R11 = ',E8.3,/,' R12 = ',E8.3,/,' R13 = ',E8.3,/,             ABP01460
     &' R14 = ',E8.3,/,' R15 = ',E8.3,/,' R16 = ',E8.3,//)              ABP01470
   30 FORMAT(' SECOND STAGE',//,                                        ABP01480
     &' C21 = ',E8.3,/,' C22 = ',E8.3,/,' R27 = ',E8.3,                 ABP01490
     &/,' R21 = ',E8.3,/,' R22 = ',E8.3,/,' R23 = ',E8.3,/,             ABP01500
     &' R24 = ',E8.3,/,' R25 = ',E8.3,/,' R26 = ',E8.3)                 ABP01510
C***********************************************************************ABP01520
C                                                                       *ABP01530
C                              SECTION 2                                *ABP01540
C                                                                       *ABP01550
C     IN THIS PORTION OF THE PROGRAM WE COMPUTE THE ANALOG TRANSFER     *ABP01560
C     FUNCTION OF THE ELLIPTIC PAND PASS FILTER.  IF THE ANALOG         *ABP01570
C     FUNCTION ALONE IS DESIRED THEN WE USE W0 FOR CALCULATIONS.        *ABP01580
C     IF THE ANALOG TRANSFER FUNCTION IS DESIRED FOR DIGITAL            *ABP01590
C     TRANSFORMATION THEN WE MUST USE THE PRE-WARPED ANALOG TO W0       *ABP01600
C     WHICH IS WODIG.                                                    ABP01610
C                                                                       *ABP01620
C***********************************************************************ABP01630
      RHO1=K1*SQRT(C/A)                                                  ABP01640
      RHO2=K2*SQRT(C/A)                                                  ABP01650
```

FILE: ABPDBP FORTRAN A1

```
      ALFA1=A1                                                          ABP01660
      ALFA2=1./A1                                                       ABP01670
      BETA1=D/E                                                         ABP01680
      BETA2=1./(D*E)                                                    ABP01690
      GAMMA1=D**2.                                                      ABP01700
      GAMMA2=1./D**2.                                                   ABP01710
      NUM1(1)=ALFA1*(WODIG**2.)                                         ABP01720
      NUM1(2)=0.                                                        ABP01730
      NUM1(3)=1.                                                        ABP01740
      DEN1(1)=GAMMA1*(WODIG**2.)                                        ABP01750
      DEN1(2)=BETA1*WODIG                                               ABP01760
      DEN1(3)=1.                                                        ABP01770
      NUM2(1)=ALFA2*(WODIG**2.)                                         ABP01780
      NUM2(2)=0.                                                        ABP01790
      NUM2(3)=1.                                                        ABP01800
      DEN2(1)=GAMMA2*(WODIG**2.)                                        ABP01810
      DEN2(2)=BETA2*WODIG                                               ABP01820
      DEN2(3)=1.                                                        ABP01830
      RHO=RHO1*RHO2                                                     ABP01840
      INUM1=3                                                           ABP01850
      INUM2=3                                                           ABP01860
      IDEN1=3                                                           ABP01870
      IDEN2=3                                                           ABP01880
      CALL PMPY(ANUM,IANUM,NUM1,INUM1,NUM2,INUM2)                       ABP01890
      CALL PMPY(ADEN,IADEN,DEN1,IDEN1,DEN2,IDEN2)                       ABP01900
      WRITE(4,110)RHO                                                   ABP01910
      WRITE(4,120)ANUM(5),ANUM(4),ANUM(3),ANUM(2),ANUM(1)               ABP01920
      DO 105 I=1,5                                                      ABP01930
         ANUM(I)=ANUM(I)*RHO                                            ABP01940
  105 CONTINUE                                                          ABP01950
      WRITE(4,130)ANUM(5),ANUM(4),ANUM(3),ANUM(2),ANUM(1)               ABP01960
      DO 106 I=1,5                                                      ABP01970
         ANUM(I)=ANUM(I)/RHO                                            ABP01980
  106 CONTINUE                                                          ABP01990
      WRITE(4,140)ADEN(5),ADEN(4),ADEN(3),ADEN(2),ADEN(1)               ABP02000
  110 FORMAT('1','SECTION 2 OUTPUT',//,                                 ABP02010
     &' ANALOG ELLIPTIC BANDPASS FILTER TRANSFER FUNCTION'              ABP02020
     &,//,' NUMERATOR COEFFICIENT = RHO = ',F9.6,/)                     ABP02030
  120 FORMAT(' NUMERATOR POLYNOMIAL (NORMALIZED)'                       ABP02040
     &,/,E12.5,' S**4 +',/,E12.5,' S**3 +'                              ABP02050
     &,/,E12.5,' S**2 +',/,E12.5,' S    +',/,E12.5,//)                  ABP02060
  130 FORMAT(' NUMERATOR POLYNOMIAL (UN-NORMALIZED)'                    ABP02070
     &,/,E12.5,' S**4 +',/,E12.5,' S**3 +'                              ABP02080
     &,/,E12.5,' S**2 +',/,E12.5,' S    +',/,E12.5,//)                  ABP02090
  140 FORMAT(' DENOMINATOR POLYNOMIAL (NORMALIZED)'                     ABP02100
     &,/,E12.5,' S**4 +',/,E12.5,                                       ABP02110
     &' S**3 +',/,E12.5,' S**2 +',/,E12.5,' S    +',/,E12.5,///)        ABP02120
C*************************************************************************ABP02130
C                                                                       *ABP02140
C                            SECTION 2A                                 *ABP02150
C                                                                       *ABP02160
C    CALCULATE THE COMPLEX POLES AND ZEROS OF THE ANALOG FILTER         *ABP02170
C                                                                       *ABP02180
C*************************************************************************ABP02190
      IPDEG = 4                                                         ABP02200
```

FILE: ABPDBP FORTRAN A1

```
      DO 1400 I=1,5                                                    ABP02210
         ANMINV(I)=ANUM(6-I)                                           ABP02220
         ADNINV(I)=ADEN(6-I)                                           ABP02230
 1400 CONTINUE                                                         ABP02240
      CALL ZRPOLY(ANMINV,IPDEG,AZERO,IERR)                             ABP02250
      CALL ZRPOLY(ADNINV,IPDEG,APOLE,IERR)                             ABP02260
      WRITE(4,1410)                                                    ABP02270
      WRITE(4,1420)AZERO(1),AZERO(2),AZERO(3),AZERO(4)                 ABP02280
      WRITE(4,1430)APOLE(1),APOLE(2),APOLE(3),APOLE(4)                 ABP02290
 1410 FORMAT(///,' SECTION 2A OUTPUT',//,                              ABP02300
     &' ANALOG FILTER COMPLEX POLES AND ZEROS',/)                      ABP02310
 1420 FORMAT(' ANALOG FILTER ZEROS',/,4(1X,E12.5,2X,E12.5,/),/)        ABP02320
 1430 FORMAT(' ANALOG FILTER POLES',/,4(1X,E12.5,2X,E12.5,/),///)      ABP02330
C*********************************************************************ABP02340
C                                                                     *ABP02350
C                            SECTION 3                                *ABP02360
C                                                                     *ABP02370
C  -   THE FOLLOWING STATEMENTS PERFORM THE BILINEAR TRANSFORMATION    *ABP02380
C      OF THE ANALOG BANDPASS TRANSFER FUNCTION COMPUTED ABOVE IN      *ABP02390
C      SECTION 2                                                      *ABP02400
C                                                                     *ABP02410
C*********************************************************************ABP02420
      IZM=5                                                            ABP02430
      IZP=5                                                            ABP02440
      IDNUM=5                                                          ABP02450
      IDDEN=5                                                          ABP02460
      ZM1(1)=-1.                                                       ABP02470
      ZM1(2)=1.                                                        ABP02480
      ZM1(3)=0.                                                        ABP02490
      ZM1(4)=0.                                                        ABP02500
      ZM1(5)=0.                                                        ABP02510
      ZM2(1)=1.                                                        ABP02520
      ZM2(2)=-2.                                                       ABP02530
      ZM2(3)=1.                                                        ABP02540
      ZM2(4)=0.                                                        ABP02550
      ZM2(5)=0.                                                        ABP02560
      ZM3(1)=-1.                                                       ABP02570
      ZM3(2)=3.                                                        ABP02580
      ZM3(3)=-3.                                                       ABP02590
      ZM3(4)=1.                                                        ABP02600
      ZM3(5)=0.                                                        ABP02610
      ZM4(1)=1.                                                        ABP02620
      ZM4(2)=-4.                                                       ABP02630
      ZM4(3)=6.                                                        ABP02640
      ZM4(4)=-4.                                                       ABP02650
      ZM4(5)=1.                                                        ABP02660
      ZP1(1)=1.                                                        ABP02670
      ZP1(2)=1.                                                        ABP02680
      ZP1(3)=0.                                                        ABP02690
      ZP1(4)=0.                                                        ABP02700
      ZP1(5)=0.                                                        ABP02710
      ZP2(1)=1.                                                        ABP02720
      ZP2(2)=2.                                                        ABP02730
      ZP2(3)=1.                                                        ABP02740
      ZP2(4)=0.                                                        ABP02750
```

```
      ZP2(5)=0.                                                          ABP02760
      ZP3(1)=1.                                                          ABP02770
      ZP3(2)=3.                                                          ABP02780
      ZP3(3)=3.                                                          ABP02790
      ZP3(4)=1.                                                          ABP02800
      ZP3(5)=0.                                                          ABP02810
      ZP4(1)=1.                                                          ABP02820
      ZP4(2)=4.                                                          ABP02830
      ZP4(3)=6.                                                          ABP02840
      ZP4(4)=4.                                                          ABP02850
      ZP4(5)=1.                                                          ABP02860
      TNCOEF=ANUM(1)*(TDIV2**4.)                                         ABP02870
      TDCOEF=ADEN(1)*(TDIV2**4.)                                         ABP02880
      DO 200 I=1,5                                                       ABP02890
         DNUM(I)=ZP4(I)*TNCOEF                                           ABP02900
         DDEN(I)=ZP4(I)*TDCOEF                                           ABP02910
  200 CONTINUE                                                           ABP02920
      TNCOEF=ANUM(2)*(TDIV2**3.)                                         ABP02930
      TDCOEF=ADEN(2)*(TDIV2**3.)                                         ABP02940
      DO 210 I=1,5                                                       ABP02950
         CALL PMPY(DTMP,IDTMP,ZM1,IZM,ZP3,IZP)                           ABP02960
         DNUM(I)=DNUM(I)+(DTMP(I)*TNCOEF)                                ABP02970
         DDEN(I)=DDEN(I)+(DTMP(I)*TDCOEF)                                ABP02980
  210 CONTINUE                                                           ABP02990
      TNCOEF=ANUM(3)*(TDIV2**2.)                                         ABP03000
      TDCOEF=ADEN(3)*(TDIV2**2.)                                         ABP03010
      DO 220 I=1,5                                                       ABP03020
         CALL PMPY(DTMP,IDTMP,ZM2,IZM,ZP2,IZP)                           ABP03030
         DNUM(I)=DNUM(I)+(DTMP(I)*TNCOEF)                                ABP03040
         DDEN(I)=DDEN(I)+(DTMP(I)*TDCOEF)                                ABP03050
  220 CONTINUE                                                           ABP03060
      TNCOEF=ANUM(4)*(TDIV2)                                             ABP03070
      TDCOEF=ADEN(4)*(TDIV2)                                             ABP03080
      DO 230 I=1,5                                                       ABP03090
         CALL PMPY(DTMP,IDTMP,ZM3,IZM,ZP1,IZP)                           ABP03100
         DNUM(I)=DNUM(I)+(DTMP(I)*TNCOEF)                                ABP03110
         DDEN(I)=DDEN(I)+(DTMP(I)*TDCOEF)                                ABP03120
  230 CONTINUE                                                           ABP03130
      TNCOEF=ANUM(5)                                                     ABP03140
      TDCOEF=ADEN(5)                                                     ABP03150
      DO 240 I=1,5                                                       ABP03160
         DNUM(I)=DNUM(I)+(ZM4(I)*TNCOEF)                                 ABP03170
         DDEN(I)=DDEN(I)+(ZM4(I)*TDCOEF)                                 ABP03180
  240 CONTINUE                                                           ABP03190
      DO 250 I=1,5                                                       ABP03200
         DNUM(I)=DNUM(I)/DDEN(5)                                         ABP03210
         DDEN(I)=DDEN(I)/DDEN(5)                                         ABP03220
  250 CONTINUE                                                           ABP03230
      RHO=RHO*DNUM(5)                                                    ABP03240
      DO 260 I=1,5                                                       ABP03250
         DNUM(I)=DNUM(I)/DNUM(5)                                         ABP03260
  260 CONTINUE                                                           ABP03270
      WRITE(4,310)RHO                                                    ABP03280
      WRITE(4,320)DNUM(5),DNUM(4),DNUM(3),DNUM(2),DNUM(1)                ABP03290
      DO 305 I=1,5                                                       ABP03300
```

FILE: ABPDBP FORTRAN A1

```
        DNUM(I)=DNUM(I)*RHO                                                 ABP03310
  305 CONTINUE                                                              ABP03320
      WRITE(4,330)DNUM(5),DNUM(4),DNUM(3),DNUM(2),DNUM(1)                   ABP03330
      WRITE(4,340)DDEN(5),DDEN(4),DDEN(3),DDEN(2),DDEN(1)                   ABP03340
  310 FORMAT('1','SECTION 3 OUTPUT',//,                                     ABP03350
     &' EQUIVALENT DIGITAL FILTER TRANSFER FUNCTION',//,                    ABP03360
     &' NUMERATOR COEFFICIENT = ',E12.5,/)                                  ABP03370
  320 FORMAT(' NUMERATOR (NORMALIZED)'                                      ABP03380
     &,/,1X,E12.5,' + ',/,1X,E12.5,' Z**-1 +',/,1X,E12.5,' Z**-2 +',        ABP03390
     &/,1X,E12.5,' Z**-3 +',/,1X,E12.5,' Z**-4',//)                         ABP03400
  330 FORMAT(' NUMERATOR (UN-NORMALIZED)'                                   ABP03410
     &,/,1X,E12.5,' + ',/,1X,E12.5,' Z**-1 +',/,1X,E12.5,' Z**-2 +',        ABP03420
     &/,1X,E12.5,' Z**-3 +',/,1X,E12.5,' Z**-4',//)                         ABP03430
  340 FORMAT(' DENOMINATOR'                                                 ABP03440
     &,/,1X,E12.5,' + ',/,1X,E12.5,' Z**-1 +',/,1X,E12.5,' Z**-2 +',        ABP03450
     &/,1X,E12.5,' Z**-3 +',/,1X,E12.5,' Z**-4',////)                       ABP03460
C*************************************************************************ABP03470
C                                                                          *ABP03480
C                              SECTION 3A                                  *ABP03490
C                                                                          *ABP03500
C    CALCULATE THE COMPLEX POLES AND ZEROS OF THE DIGITAL FILTER           *ABP03510
C    AND THE RADIUS OF THE COMPLEX POLE AND ZERO VECTORS RELATIVE          *ABP03520
C    TO THE UNIT CIRCLE.                                                   *ABP03530
C                                                                          *ABP03540
C*************************************************************************ABP03550
      IPDEG = 4                                                             ABP03560
      DO 400 I=1,5                                                          ABP03570
         DNMINV(I)=DNUM(6-I)                                                ABP03580
         DDNINV(I)=DDEN(6-I)                                                ABP03590
  400 CONTINUE                                                              ABP03600
      CALL ZRPOLY(DNMINV,IPDEG,DZERO,IERR)                                  ABP03610
      CALL ZRPOLY(DDNINV,IPDEG,DPOLE,IERR)                                  ABP03620
      DO 404 I=1,4                                                          ABP03630
         RZ(I)=SQRT((REAL(DZERO(I)))**2.+(AIMAG(DZERO(I)))**2.)             ABP03640
         RP(I)=SQRT((REAL(DPOLE(I)))**2.+(AIMAG(DPOLE(I)))**2.)             ABP03650
  404 CONTINUE                                                              ABP03660
      WRITE(4,410)                                                          ABP03670
      WRITE(4,420)DZERO(1),RZ(1),DZERO(2),RZ(2),DZERO(3),RZ(3),             ABP03680
     &DZERO(4),RZ(4)                                                        ABP03690
      WRITE(4,430)DPOLE(1),RP(1),DPOLE(2),RP(2),DPOLE(3),RP(3),             ABP03700
     &DPOLE(4),RP(4)                                                        ABP03710
  410 FORMAT(' SECTION 3A OUTPUT',//,                                       ABP03720
     &' DIGITAL FILTER COMPLEX POLES AND ZEROS AND RADIUS',//)              ABP03730
  420 FORMAT(' ZERO LOCATIONS (REAL,IMAG) AND RADIUS',/,                    ABP03740
     &4(E12.5,2X,E12.5,2X,E12.5,/),/)                                       ABP03750
  430 FORMAT(' POLE LOCATIONS (REAL,IMAG) AND RADIUS',/,                    ABP03760
     &4(E12.5,2X,E12.5,2X,E12.5,/),///)                                     ABP03770
C*************************************************************************ABP03780
C                                                                          *ABP03790
C                              SECTION 4                                   *ABP03800
C                                                                          *ABP03810
C    COMPUTE THE POLYNOMIAL COEFFICIENTS OF QUADRATIC FACTORS FOR          *ABP03820
C    THE POLES AND ZEROS PREVIOUSLY DETERMINED.                            *ABP03830
C                                                                          *ABP03840
C*************************************************************************ABP03850
```

```
      I4=4                                                              ABP03860
      I2=2                                                              ABP03870
      DO 500 I=1,4                                                      ABP03880
         DZ4(I)=CMPLX(REAL(DZERO(I)),AIMAG(DZERO(I)))                   ABP03890
         DP4(I)=CMPLX(REAL(DPOLE(I)),AIMAG(DPOLE(I)))                   ABP03900
  500 CONTINUE                                                          ABP03910
      DO 502 I=1,2                                                      ABP03920
         DZ21(I)=DZ4(I)                                                 ABP03930
         DP21(I)=DP4(I)                                                 ABP03940
  502 CONTINUE                                                          ABP03950
      DO 504 I=3,4,1                                                    ABP03960
         DZ22(I-2)=DZ4(I)                                               ABP03970
         DP22(I-2)=DP4(I)                                               ABP03980
  504 CONTINUE                                                          ABP03990
      CALL MAKPOL(I4,DZ4,DN4)                                           ABP04000
      CALL MAKPOL(I4,DP4,DD4)                                           ABP04010
      DN45(5)=CMPLX(1.,0.)                                              ABP04020
      DD45(5)=CMPLX(1.,0.)                                              ABP04030
      DO 512 I=1,4                                                      ABP04040
         DN45(I)=DN4(I)                                                 ABP04050
         DD45(I)=DD4(I)                                                 ABP04060
  512 CONTINUE                                                          ABP04070
      WRITE(4,514)                                                      ABP04080
  514 FORMAT('1','SECTION 4 OUTPUT',//,                                 ABP04090
     &' REASSEMBLE COEFFICIENTS FROM POLES AND ZEROS',//)               ABP04100
      WRITE(4,515)                                                      ABP04110
      WRITE(4,520)DN45(5),DN45(4),DN45(3),DN45(2),DN45(1)               ABP04120
      WRITE(4,530)DD45(5),DD45(4),DD45(3),DD45(2),DD45(1)               ABP04130
  515 FORMAT(/' SINGLE FOURTH ORDER TRANSFER FUNCTION COEFFICIENTS',//) ABP04140
  520 FORMAT(' FOURTH ORDER NUMERATOR COEFFICIENTS (NORMALIZED)',/,      ABP04150
     &5(1X,E12.5,2X,E12.5,/))                                           ABP04160
  530 FORMAT(' FOURTH ORDER DENOMINATOR COEFFICIENTS (NORMALIZED)',/,   ABP04170
     &5(1X,E12.5,2X,E12.5,/),//)                                        ABP04180
      CALL MAKPOL(I2,DZ21,DN21)                                         ABP04190
      CALL MAKPOL(I2,DP21,DD21)                                         ABP04200
      DN213(3)=CMPLX(1.,0.)                                             ABP04210
      DD213(3)=CMPLX(1.,0.)                                             ABP04220
      DO 612 I=1,2                                                      ABP04230
         DN213(I)=DN21(I)                                               ABP04240
         DD213(I)=DD21(I)                                               ABP04250
  612 CONTINUE                                                          ABP04260
      WRITE(4,614)                                                      ABP04270
  614 FORMAT(' CASCADED SECOND ORDER TRANSFER FUNCTION COEFFICIENTS',/) ABP04280
      WRITE(4,615)                                                      ABP04290
      WRITE(4,620)DN213(3),DN213(2),DN213(1)                            ABP04300
      WRITE(4,630)DD213(3),DD213(2),DD213(1)                            ABP04310
  615 FORMAT(/' FIRST STAGE QUADRATIC FUNCTION COEFFICIENTS',/)         ABP04320
  620 FORMAT(' COMPLEX (REAL,IMAG) NUMERATOR COEFFICIENTS',/,           ABP04330
     &1X,E12.5,2X,E12.5,                                                ABP04340
     &'  Z**2 +',/,1X,E12.5,2X,E12.5,'  Z +',/,1X,E12.5,2X,E12.5,/)     ABP04350
  630 FORMAT(' COMPLEX (REAL,IMAG) DENOMINATOR COEFFICIENTS',/,         ABP04360
     &1X,E12.5,2X,E12.5,                                                ABP04370
     &'  Z**2 +',/,1X,E12.5,2X,E12.5,'  Z +',/,1X,E12.5,2X,E12.5,/)     ABP04380
      CALL MAKPOL(I2,DZ22,DN22)                                         ABP04390
      CALL MAKPOL(I2,DP22,DD22)                                         ABP04400
```

FILE: ABPDBP FORTRAN A1

```
      DN223(3)=CMPLX(1.,0.)                                              ABP04410
      DD223(3)=CMPLX(1.,0.)                                              ABP04420
      DO 712 I=1,2                                                       ABP04430
         DN223(I)=DN22(I)                                                ABP04440
         DD223(I)=DD22(I)                                                ABP04450
  712 CONTINUE                                                           ABP04460
      WRITE(4,715)                                                       ABP04470
      WRITE(4,720)DN223(3),DN223(2),DN223(1)                             ABP04480
      WRITE(4,730)DD223(3),DD223(2),DD223(1)                             ABP04490
  715 FORMAT(/' SECOND STAGE QUADRATIC FUNCTION COEFFICIENTS',//)        ABP04500
  720 FORMAT(' COMPLEX (REAL,IMAG) NUMERATOR COEFFICIENTS',/,            ABP04510
     &1X,E12.5,2X,E12.5,                                                 ABP04520
     &'  Z**2 +',/,1X,E12.5,2X,E12.5,'  Z +',/,1X,E12.5,2X,E12.5,/)     ABP04530
  730 FORMAT(' COMPLEX (REAL,IMAG) DENOMINATOR COEFFICIENTS',/,          ABP04540
     &1X,E12.5,2X,E12.5,                                                 ABP04550
     &'  Z**2 +',/,1X,E12.5,2X,E12.5,'  Z +',/,1X,E12.5,2X,E12.5,/)     ABP04560
      STOP                                                               ABP04570
      END                                                                ABP04580
C                                                                        ABP04590
C     ..................................................................ABP04600
C                                                                        ABP04610
C        SUBROUTINE PMPY                                                 ABP04620
C                                                                        ABP04630
C        PURPOSE                                                         ABP04640
C           MULTIPLY TWO POLYNOMIALS                                     ABP04650
C                                                                        ABP04660
C        USAGE                                                           ABP04670
C           CALL PMPY(Z,IDIMZ,X,IDIMX,Y,IDIMY)                           ABP04680
C                                                                        ABP04690
C        DESCRIPTION OF PARAMETERS                                       ABP04700
C           Z      - VECTOR OF RESULTANT COEFFICIENTS, ORDERED FROM      ABP04710
C                    SMALLEST TO LARGEST POWER                          ABP04720
C           IDIMZ  - DIMENSION OF Z (CALCULATED)                         ABP04730
C           X      - VECTOR OF COEFFICIENTS FOR FIRST POLYNOMIAL, ORDEREDABP04740
C                    FROM SMALLEST TO LARGEST POWER                     ABP04750
C           IDIMX  - DIMENSION OF X (DEGREE IS IDIMX-1)                  ABP04760
C           Y      - VECTOR OF COEFFICIENTS FOR SECOND POLYNOMIAL,       ABP04770
C                    ORDERED FROM SMALLEST TO LARGEST POWER             ABP04780
C           IDIMY  - DIMENSION OF Y (DEGREE IS IDIMY-1)                  ABP04790
C                                                                        ABP04800
C        REMARKS                                                         ABP04810
C           Z CANNOT BE IN THE SAME LOCATION AS X                        ABP04820
C           Z CANNOT BE IN THE SAME LOCATION AS Y                        ABP04830
C                                                                        ABP04840
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                   ABP04850
C           NONE                                                         ABP04860
C                                                                        ABP04870
C        METHOD                                                          ABP04880
C           DIMENSION OF Z IS CALCULATED AS IDIMX+IDIMY-1                ABP04890
C           THE COEFFICIENTS OF Z ARE CALCULATED AS SUM OF PRODUCTS      ABP04900
C           OF COEFFICIENTS OF X AND Y , WHOSE EXPONENTS ADD UP TO THE   ABP04910
C           CORRESPONDING EXPONENT OF Z.                                 ABP04920
C                                                                        ABP04930
C     ..................................................................ABP04940
C                                                                        ABP04950
```

FILE: ABPDBP FORTRAN A1

```
      SUBROUTINE PMPY(Z,IDIMZ,X,IDIMX,Y,IDIMY)                          ABP04960
      DIMENSION Z(1),X(1),Y(1)                                          ABP04970
C                                                                       ABP04980
      IF(IDIMX*IDIMY)10,10,20                                           ABP04990
   10 IDIMZ=0                                                           ABP05000
      GO TO 50                                                          ABP05010
   20 IDIMZ=IDIMX+IDIMY-1                                               ABP05020
      DO 30 I=1,IDIMZ                                                   ABP05030
   30 Z(I)=0.                                                           ABP05040
      DO 40 I=1,IDIMX                                                   ABP05050
      DO 40 J=1,IDIMY                                                   ABP05060
      K=I+J-1                                                           ABP05070
   40 Z(K)=X(I)*Y(J)+Z(K)                                               ABP05080
   50 RETURN                                                            ABP05090
      END                                                               ABP05100
C                                                                       ABP05110
C-----------------------------------------------------------------------ABP05120
C                                                                       ABP05130
C     SUBROUTINE MAKPOL                                                 ABP05140
C                                                                       ABP05150
C     PURPOSE                                                           ABP05160
C        TO COMPUTE THE COMPLEX COEFFICIENTS OF AN N-TH DEGREE POLY-    ABP05170
C        NOMIAL GIVEN N COMPLEX ROOTS OF THE POLYNOMIAL                 ABP05180
C                                                                       ABP05190
C     USAGE                                                             ABP05200
C        CALL MAKPOL(N,R,C)                                             ABP05210
C                                                                       ABP05220
C     DESCRIPTION OF PARAMETERS                                         ABP05230
C        N - NUMBER OF ROOTS GIVEN AND DEGREE OF POLYNOMIAL.  THE       ABP05240
C            COEFFICIENT OF THE HIGHEST POWER OF THE UNKNOWN IS ALWAYS ABP05250
C            UNITY, AND IS NOT COMPUTED BY "MAKPOL".                    ABP05260
C        R - DOUBLE PRECISION COMPLEX ARRAY CONTAINING THE COMPLEX      ABP05270
C            ROOTS                                                      ABP05280
C        C - DOUBLE PRECISION COMPLEX ARRAY CONTAINING THE COMPLEX      ABP05290
C            COEFFICIENTS                                               ABP05300
C                                                                       ABP05310
C     REMARKS                                                           ABP05320
C        ARRAYS R AND C ARE TYPED COMPLEX*16                            ABP05330
C                                                                       ABP05340
      SUBROUTINE MAKPOL(N,R,C)                                          ABP05350
      COMPLEX*16 R(N),C(N)                                              ABP05360
      IF(N.LE.0) RETURN                                                 ABP05370
      DO 10 I=1,N                                                       ABP05380
   10 C(I)=R(I)                                                         ABP05390
      K=N                                                               ABP05400
      M=N-1                                                             ABP05410
      DO 20 L=1,M                                                       ABP05420
      DO 30 I=2,K                                                       ABP05430
   30 C(I)=C(I)+C(I-1)                                                  ABP05440
      K=K-1                                                             ABP05450
      DO 20 I=1,K                                                       ABP05460
      J=I+L                                                             ABP05470
      C(I)=R(J)*C(I)                                                    ABP05480
   20 CONTINUE                                                          ABP05490
      K=N/2                                                             ABP05500
```

FILE: ABPDBP FORTRAN A1

```
      K=2*K/(N-K)                                                    ABP05510
      DO 40 I=K,N,2                                                  ABP05520
   40 C(I)=-C(I)                                                     ABP05530
      RETURN                                                         ABP05540
      END                                                            ABP05550
```

```
ABPDBP OUTPUT

SECTION 1 OUTPUT

INPUT AND DERIVED PARAMETERS FOR FURTHER CALCULATIONS

A = 21.164001
B =  0.787152
C =  0.842554
D =  1.035090
E = 30.507828
A1 =  1.463835
F0 =   600.000
W0 =  3769.911
Q = 12.000000
K =  1.000000
K1 =  1.000000
K2  1.000000
W0(DIG) =  3712.266
F0(DIG) =   590.825


ELLIPTIC ANALOG BPF COMPONENT VALUES

FIRST STAGE

C11 = .996E-08
C12 = .995E-08
R17 = .267E+05
R11 = .393E+07
R12 = .785E+06
R13 = .257E+05
R14 = .134E+06
R15 = .945E+05
R16 = .258E+05


SECOND STAGE

C21 = .103E-07
C22 = .103E-07
R27 = .267E+05
R21 = .408E+07
R22 = .813E+06
R23 = .267E+05
R24 = .134E+06
R25 = .182E+06
R26 = .266E+05
```

```
SECTION 2 OUTPUT

ANALOG ELLIPTIC BANDPASS FILTER TRANSFER FUNCTION

NUMERATOR COEFFICIENT = RHO =  0.039811

NUMERATOR POLYNOMIAL (NORMALIZED)
0.10000E+01 S**4 +
0.0         S**3 +
0.29587E+08 S**2 +
0.0         S    +
0.18991E+15


NUMERATOR POLYNOMIAL (UN-NORMALIZED)
0.39811E-01 S**4 +
0.0         S**3 +
0.11779E+07 S**2 +
0.0         S    +
0.75606E+13

DENOMINATOR POLYNOMIAL (NORMALIZED)
0.10000E+01 S**4 +
0.24351E+03 S**3 +
0.27642E+08 S**2 +
0.33558E+10 S    +
0.18991E+15



SECTION 2A OUTPUT

ANALOG FILTER COMPLEX POLES AND ZEROS

ANALOG FILTER ZEROS
-0.45776E-03   0.30683E+04
-0.45776E-03  -0.30683E+04
 0.45776E-03   0.44914E+04
 0.45776E-03  -0.44914E+04


ANALOG FILTER POLES
-0.58188E+02   0.35859E+04
-0.58188E+02  -0.35859E+04
-0.63567E+02   0.38421E+04
-0.63567E+02  -0.38421E+04
```

```
SECTION 3 OUTPUT

EQUIVALENT DIGITAL FILTER TRANSFER FUNCTION

NUMERATOR COEFFICIENT =  0.39516E-01

NUMERATOR (NORMALIZED)
 0.10000E+01 +
-0.36279E+01 Z**-1 +
 0.52861E+01 Z**-2 +
-0.36279E+01 Z**-3 +
 0.10000E+01 Z**-4


NUMERATOR (UN-NORMALIZED)
 0.39516E-01 +
-0.14336E+00 Z**-1 +
 0.20888E+00 Z**-2 +
-0.14336E+00 Z**-3 +
 0.39516E-01 Z**-4


DENOMINATOR
 0.10000E+01 +
-0.36251E+01 Z**-1 +
 0.52586E+01 Z**-2 +
-0.35768E+01 Z**-3 +
 0.97353E+00 Z**-4



SECTION 3A OUTPUT

DIGITAL FILTER COMPLEX POLES AND ZEROS AND RADIUS


ZERO LOCATIONS (REAL,IMAG) AND RADIUS
0.87496E+00   0.48595E+00   0.10009E+01
0.87496E+00  -0.48595E+00   0.10009E+01
0.93896E+00   0.34135E+00   0.99909E+00
0.93896E+00  -0.34135E+00   0.99909E+00


POLE LOCATIONS (REAL,IMAG) AND RADIUS
0.90781E+00   0.40813E+00   0.99533E+00
0.90781E+00  -0.40813E+00   0.99533E+00
0.90475E+00   0.40493E+00   0.99123E+00
0.90475E+00  -0.40493E+00   0.99123E+00
```

```
SECTION 4 OUTPUT

REASSEMBLE COEFFICIENTS FROM POLES AND ZEROS


SINGLE FOURTH ORDER TRANSFER FUNCTION COEFFICIENTS


FOURTH ORDER NUMERATOR COEFFICIENTS (NORMALIZED)
 0.10000E+01    0.0
-0.36279E+01    0.0
 0.52861E+01    0.0
-0.36279E+01    0.0
 0.99988E+00    0.0

FOURTH ORDER DENOMINATOR COEFFICIENTS (NORMALIZED)
 0.10000E+01    0.0
-0.36251E+01    0.0
 0.52586E+01    0.0
-0.35766E+01   -0.18041E-15
 0.97339E+00    0.0


CASCADED SECOND ORDER TRANSFER FUNCTION COEFFICIENTS


FIRST STAGE QUADRATIC FUNCTION COEFFICIENTS

COMPLEX (REAL,IMAG) NUMERATOR COEFFICIENTS
 0.10000E+01    0.0           Z**2 +
-0.17499E+01    0.0           Z +
 0.10017E+01    0.0

COMPLEX (REAL,IMAG) DENOMINATOR COEFFICIENTS
 0.10000E+01    0.0           Z**2 +
-0.18156E+01    0.0           Z +
 0.99068E+00    0.0


SECOND STAGE QUADRATIC FUNCTION COEFFICIENTS


COMPLEX (REAL,IMAG) NUMERATOR COEFFICIENTS
 0.10000E+01    0.0           Z**2 +
-0.18779E+01    0.0           Z +
 0.99817E+00    0.0

COMPLEX (REAL,IMAG) DENOMINATOR COEFFICIENTS
 0.10000E+01    0.0           Z**2 +
-0.18095E+01    0.0           Z +
 0.98255E+00    0.0
```

FILE: ABPFR FORTRAN A1

```
C*************************************************************************ABP00010
C                                                                         *ABP00020
C                              APPENDIX D                                 *ABP00030
C                                                                         *ABP00040
C                         FORTRAN PROGRAM ABPFR                           *ABP00050
C                                                                         *ABP00060
C                                                                         *ABP00070
C     PROGRAM TO PLOT ANALOG BAND-PASS FILTER FREQUENCY AND PHASE         *ABP00080
C     RESPONSE OF THE ELLIPTIC FILTER TRANSFER FUNCTION                   *ABP00090
C                                                                         *ABP00100
C*************************************************************************ABP00110
C     TYPE DECLARATIONS                                                    ABP00120
      IMPLICIT REAL(A-H,O-Z),INTEGER(I-N)                                  ABP00130
      REAL W(1001),HMAG(1001),HMAGN(1001),HMAGDB(1001),WX,F(1001)          ABP00140
      REAL A(5),B(5),HPHASE(1001)                                          ABP00150
      COMPLEX S,H                                                          ABP00160
C     NORMALIZED ANALOG TRANSFER FUNCTION COEFFICIENTS                     ABP00170
      A(1) = 1.                                                            ABP00180
      A(2) = 0.0                                                           ABP00190
      A(3) = 0.30513E+8                                                    ABP00200
      A(4) = 0.0                                                           ABP00210
      A(5) = 0.20198E+15                                                   ABP00220
      B(1) = 1.0                                                           ABP00230
      B(2) = 0.24729E+3                                                    ABP00240
      B(3) = 0.28507E+8                                                    ABP00250
      B(4) = 0.35145E+10                                                   ABP00260
      B(5) = 0.20198E+15                                                   ABP00270
C     CONSTANTS                                                            ABP00280
      PI = 3.1415927                                                       ABP00290
C     EVALUATE MAGNITUDE AND PHASE OF H(JW)                                ABP00300
      DO 10 I = 1,1001                                                     ABP00310
         F(I) = FLOAT(I-1)                                                 ABP00320
         W(I) = (2.*PI*F(I))                                               ABP00330
         S = CMPLX(0.,W(I))                                                ABP00340
         H = (A(1)*(S**4)+A(2)*(S**3)+A(3)*(S**2)+A(4)*S+A(5))/            ABP00350
     &(B(1)*(S**4)+B(2)*(S**3)+B(3)*(S**2)+B(4)*S+B(5))                    ABP00360
         HMAG(I) = CABS(H)                                                 ABP00370
         X = REAL(H)                                                       ABP00380
         Y = AIMAG(H)                                                      ABP00390
         HPHASE(I) = ATAN(Y/X)*180./PI                                     ABP00400
   10 CONTINUE                                                             ABP00410
C     NORMALIZE MAGNITUDE                                                  ABP00420
      AMAX = 0.0                                                           ABP00430
      DO 20 I = 1,1001                                                     ABP00440
         IF(HMAG(I).GT.AMAX) AMAX = HMAG(I)                                ABP00450
   20 CONTINUE                                                             ABP00460
      DO 30 I = 1,1001                                                     ABP00470
         HMAGN(I) = HMAG(I)/AMAX                                           ABP00480
         HMAGDB(I) = 20.0 * ALOG10(HMAG(I))                                ABP00490
   30 CONTINUE                                                             ABP00500
      DO 40 I = 1,1001                                                     ABP00510
         WRITE (4,50)I,F(I),W(I),HMAG(I),HMAGN(I),HMAGDB(I),HPHASE(I)      ABP00520
   50 FORMAT(I4,6(1X,E10.3))                                               ABP00530
   40 CONTINUE                                                             ABP00540
C------------------------------------------------------------------------- ABP00550
```

FILE: ABPFR   FORTRAN A1

```
C GRAPHICS PARAMETERS FOR MAGNITUDE VS FREQUENCY                        ABP00560
C---------------------------------------------------------------------- ABP00570
      CALL LRGBUF                                                       ABP00580
      CALL COMPRS                                                       ABP00590
C     CALL TEK618                                                       ABP00600
C     CALL VRSTEC(0,0,0)                                                ABP00610
C .....     SETUP THE PLOTTING AREA                                     ABP00620
      CALL PAGE (11.0,8.5)                                              ABP00630
      CALL NOBRDR                                                       ABP00640
      CALL AREA2D(9.0,6.5)                                              ABP00650
C .....  LABEL THE X & Y AXES                                           ABP00660
      CALL XNAME('FREQUENCY (HZ)$',100)                                 ABP00670
      CALL YNAME('AMPLITUDE$',100)                                      ABP00680
      CALL HEADIN('ANALOG ELLIPTIC BPF FREQUENCY RESPONSE$',100,1.6,2)  ABP00690
      CALL HEADIN('AMPLITUDE VS FREQ (F0=600 HZ)$',100,1.,2)            ABP00700
C .....   DEFINE THE AXES                                              ABP00710
      CALL GRAF(0.0,'SCALE',1000.,-0.5,'SCALE',+1.5)                    ABP00720
C..... DRAW THE CURVES                                                  ABP00730
C     CALL THKCRV(0.02)                                                 ABP00740
      CALL MARKER(15)                                                   ABP00750
      CALL CURVE(F,HMAGN,1001,0)                                        ABP00760
C..... TERMINATE THIS PLOT                                              ABP00770
      CALL ENDPL(0)                                                     ABP00780
C---------------------------------------------------------------------- ABP00790
C GRAPHICS PARAMETERS FOR MAGNITUDE IN DBS VS FREQUENCY                 ABP00800
C---------------------------------------------------------------------- ABP00810
      CALL LRGBUF                                                       ABP00820
      CALL COMPRS                                                       ABP00830
C     CALL TEK618                                                       ABP00840
C     CALL VRSTEC(0,0,0)                                                ABP00850
C .....     SETUP THE PLOTTING AREA                                     ABP00860
      CALL PAGE (11.0,8.5)                                              ABP00870
      CALL NOBRDR                                                       ABP00880
      CALL AREA2D(9.0,6.5)                                              ABP00890
C .....  LABEL THE X & Y AXES                                           ABP00900
      CALL XNAME('FREQUENCY (HZ)$',100)                                 ABP00910
      CALL YNAME('AMPLITUDE (DB)$',100)                                 ABP00920
      CALL HEADIN('ANALOG ELLIPTIC BPF FREQUENCY RESPONSE$',100,1.6,2)  ABP00930
      CALL HEADIN('AMPLITUDE (DB) VS FREQ (F0=600 HZ)$',100,1.,2)       ABP00940
C .....   DEFINE THE AXES                                              ABP00950
      CALL GRAF(0.0,'SCALE',1000.,-60.0,'SCALE',30.0)                   ABP00960
C..... DRAW THE CURVES                                                  ABP00970
C     CALL THKCRV(0.02)                                                 ABP00980
      CALL MARKER(15)                                                   ABP00990
      CALL CURVE(F,HMAGDB,1001,0)                                       ABP01000
C..... TERMINATE THIS PLOT                                              ABP01010
      CALL ENDPL(0)                                                     ABP01020
C---------------------------------------------------------------------- ABP01030
C GRAPHICS PARAMETERS FOR PHASE VS FREQUENCY                            ABP01040
C---------------------------------------------------------------------- ABP01050
      CALL LRGBUF                                                       ABP01060
      CALL COMPRS                                                       ABP01070
C     CALL TEK618                                                       ABP01080
C     CALL VRSTEC(0,0,0)                                                ABP01090
C .....     SETUP THE PLOTTING AREA                                     ABP01100
```

FILE: ABPFR FORTRAN A1

```
      CALL PAGE (11.0,8.5)                                                    ABP01110
      CALL NOBRDR                                                             ABP01120
      CALL AREA2D(9.0,6.5)                                                    ABP01130
C .....  LABEL THE X & Y AXES                                                 ABP01140
      CALL XNAME('FREQUENCY (HZ)$',100)                                       ABP01150
      CALL YNAME('PHASE (DEGREES)$',100)                                      ABP01160
      CALL HEADIN('ANALOG ELLIPTIC BPF PHASE RESPONSE$',100,1.6,2)            ABP01170
      CALL HEADIN ('PHASE VS FREQ (F0=600 HZ)$',100,1.,2)                     ABP01180
C .....   DEFINE THE AXES                                                     ABP01190
      CALL GRAF(0.0,'SCALE',1000.,-100.,'SCALE',100.)                         ABP01200
C..... DRAW THE CURVES                                                        ABP01210
C     CALL THKCRV(0.02)                                                       ABP01220
      CALL MARKER(15)                                                         ABP01230
      CALL CURVE(F,HPHASE,1001,0)                                             ABP01240
C..... TERMINATE THIS PLOT                                                    ABP01250
      CALL ENDPL(0)                                                           ABP01260
      CALL DONEPL                                                             ABP01270
      STOP                                                                    ABP01280
      END                                                                     ABP01290
```

FILE: DBPFR FORTRAN A1

```
C*************************************************************************DBP00010
C                                                                         *DBP00020
C                              APPENDIX E                                 *DBP00030
C                                                                         *DBP00040
C                         FORTRAN PROGRAM DBPFR                           *DBP00050
C                                                                         *DBP00060
C                                                                         *DBP00070
C    PROGRAM TO PLOT DIGITAL BAND-PASS FILTER FREQUENCY AND PHASE         *DBP00080
C    RESPONSE OF THE ELLIPTIC FILTER TRANSFER FUNCTION                    *DBP00090
C                                                                         *DBP00100
C*************************************************************************DBP00110
C    TYPE DECLARATIONS                                                     DBP00120
      IMPLICIT REAL(A-H,O-Z),INTEGER(I-N)                                  DBP00130
      REAL OMEGA(201),HMAG(201),HPHASE(201),HMAGN(201),HMAGDB(201)         DBP00140
      REAL F(201),FREQ(201),FS,FSDIV2,TS                                   DBP00150
      COMPLEX Z,H                                                          DBP00160
C    NORMALIZED TRANSFER FUNCTION COEFFICIENTS                             DBP00170
      A0 = 1.                                                              DBP00180
      A1 =-3.6279                                                          DBP00190
      A2 = 5.2861                                                          DBP00200
      A3 = A2                                                              DBP00210
      A4 = A1                                                              DBP00220
      B0 = 1.                                                              DBP00230
      B1 =-3.6251                                                          DBP00240
      B2 = 5.2586                                                          DBP00250
      B3 =-3.5768                                                          DBP00260
      B4 = 0.97353                                                         DBP00270
C    CONSTANTS                                                             DBP00280
      PI = 3.1415927                                                       DBP00290
      FS = 6.666E6/(4.*192.)                                               DBP00300
      TS=1./FS                                                             DBP00310
      FSDIV2 = FS/2.                                                       DBP00320
C    EVALUATE MAGNITUDE AND PHASE OF H(EXP(J*OMEGA*T))                     DBP00330
      DO 10 I = 1,201                                                      DBP00340
         F(I) = FLOAT(I-1)                                                 DBP00350
         FREQ(I) = 6.*F(I)                                                 DBP00360
         OMEGA(I) = (2.*PI*FREQ(I)*TS)                                     DBP00370
         Z = CMPLX(COS(OMEGA(I)),SIN(OMEGA(I)))                            DBP00380
         H = (A0+A1*Z**(-1)+A2*Z**(-2)+A3*Z**(-3)+A4*Z**(-4))/(B0+         DBP00390
     &B1*Z**(-1)+B2*Z**(-2)+B3*Z**(-3)+B4*Z**(-4))                         DBP00400
         HMAG(I) = CABS(H)                                                 DBP00410
         X = REAL(H)                                                       DBP00420
         Y = AIMAG(H)                                                      DBP00430
         HPHASE(I) = ATAN(Y/X)*180./PI                                     DBP00440
   10 CONTINUE                                                             DBP00450
C    NORMALIZE MAGNITUDE                                                   DBP00460
      AMAX = 0.0                                                           DBP00470
      DO 20 I = 1,201                                                      DBP00480
         IF(HMAG(I).GT.AMAX) AMAX = HMAG(I)                                DBP00490
   20 CONTINUE                                                             DBP00500
      DO 30 I = 1,201                                                      DBP00510
         HMAGN(I) = HMAG(I)/AMAX                                           DBP00520
         HMAGDB(I) = 20.0 * ALOG10(HMAG(I))                                DBP00530
   30 CONTINUE                                                             DBP00540
C------------------------------------------------------------------------- DBP00550
```

FILE: DBPFR FORTRAN A1

```
C GRAPHICS PARAMETERS FOR MAGNITUDE VS FREQUENCY (IN HZ)                    DBP00560
C-------------------------------------------------------------------------- DBP00570
      CALL LRGBUF                                                           DBP00580
C     CALL COMPRS                                                           DBP00590
      CALL TEK618                                                           DBP00600
C     CALL VRSTEC(0,0,0)                                                    DBP00610
C .....     SETUP THE PLOTTING AREA                                         DBP00620
      CALL PAGE (11.0,8.5)                                                  DBP00630
      CALL NOBRDR                                                           DBP00640
      CALL AREA2D(9.0,6.5)                                                  DBP00650
C .....  LABEL THE X & Y AXES                                              DBP00660
      CALL XNAME('FREQUENCY (HZ)$',100)                                     DBP00670
      CALL YNAME('AMPLITUDE$',100)                                          DBP00680
      CALL HEADIN('DIGITAL ELLIPTIC BPF FREQUENCY RESPONSE$',100,1.6,2)     DBP00690
      CALL HEADIN('NORMALIZED AMPLITUDE VS FREQ (F0=590 HZ)$',100,1.,2)     DBP00700
C .....   DEFINE THE AXES                                                  DBP00710
      CALL GRAF(0.0,'SCALE',1200.,-0.5,'SCALE',1.5)                         DBP00720
C..... DRAW THE CURVES                                                      DBP00730
      CALL THKCRV(0.02)                                                     DBP00740
      CALL MARKER(15)                                                       DBP00750
      CALL CURVE(FREQ,HMAGN,201,0)                                          DBP00760
C..... TERMINATE THIS PLOT                                                  DBP00770
      CALL ENDPL(0)                                                         DBP00780
C-------------------------------------------------------------------------- DBP00790
C GRAPHICS PARAMETERS FOR MAGNITUDE (IN DB) VS FREQUENCY (IN HZ)            DBP00800
C-------------------------------------------------------------------------- DBP00810
      CALL LRGBUF                                                           DBP00820
C     CALL COMPRS                                                           DBP00830
      CALL TEK618                                                           DBP00840
C     CALL VRSTEC(0,0,0)                                                    DBP00850
C .....     SETUP THE PLOTTING AREA                                         DBP00860
      CALL PAGE (11.0,8.5)                                                  DBP00870
      CALL NOBRDR                                                           DBP00880
      CALL AREA2D(9.0,6.5)                                                  DBP00890
C .....  LABEL THE X & Y AXES                                              DBP00900
      CALL XNAME('FREQUENCY (HZ)$',100)                                     DBP00910
      CALL YNAME('AMPLITUDE (DB)$',100)                                     DBP00920
      CALL HEADIN('DIGITAL ELLIPTIC BPF FREQUENCY RESPONSE$',100,1.6,2)     DBP00930
      CALL HEADIN('AMPLITUDE (DB) VS FREQ (F0=590 HZ)$',100,1.,2)           DBP00940
C .....   DEFINE THE AXES                                                  DBP00950
      CALL GRAF(0.0,'SCALE',1200.,20.0,'SCALE',100.0)                       DBP00960
C..... DRAW THE CURVES                                                      DBP00970
      CALL THKCRV(0.02)                                                     DBP00980
      CALL MARKER(15)                                                       DBP00990
      CALL CURVE(FREQ,HMAGDB,201,0)                                         DBP01000
C..... TERMINATE THIS PLOT                                                  DBP01010
      CALL ENDPL(0)                                                         DBP01020
C-------------------------------------------------------------------------- DBP01030
C GRAPHICS PARAMETERS FOR PHASE VS FREQUENCY                                DBP01040
C-------------------------------------------------------------------------- DBP01050
      CALL LRGBUF                                                           DBP01060
      CALL COMPRS                                                           DBP01070
C     CALL TEK618                                                           DBP01080
C     CALL VRSTEC(0,0,0)                                                    DBP01090
C .....     SETUP THE PLOTTING AREA                                         DBP01100
```

FILE: DBPFR FORTRAN A1

```
      CALL PAGE (11.0,8.5)                                                  DBP01110
      CALL NOBRDR                                                           DBP01120
      CALL AREA2D(9.0,6.5)                                                  DBP01130
C .....  LABEL THE X & Y AXES                                               DBP01140
      CALL XNAME('FREQUENCY (HZ)$',100)                                     DBP01150
      CALL YNAME('PHASE (DEGREES)$',100)                                    DBP01160
      CALL HEADIN('DIGITAL ELLIPTIC BPF PHASE RESPONSE$',100,1.6,2)         DBP01170
      CALL HEADIN('PHASE (DEGREES) VS FREQ (F0=590 HZ)$',100,1.,2)          DBP01180
C .....   DEFINE THE AXES                                                   DBP01190
      CALL GRAF(0.0,'SCALE',1200.,-100.,'SCALE',100.)                       DBP01200
C.....  DRAW THE CURVES                                                     DBP01210
      CALL THKCRV(0.02)                                                     DBP01220
      CALL MARKER(15)                                                       DBP01230
      CALL CURVE(FREQ,HPHASE,201,0)                                         DBP01240
C.....  TERMINATE THIS PLOT                                                 DBP01250
      CALL ENDPL(0)                                                         DBP01260
      CALL DONEPL                                                           DBP01270
      STOP                                                                  DBP01280
      END                                                                   DBP01290
```

FILE: S221 FORTRAN A1

```
C*************************************************************************S2200010
C                                                                         *S2200020
C                              APPENDIX F                                 *S2200030
C                                                                         *S2200040
C                         FORTRAN PROGRAM S221                            *S2200050
C                                                                         *S2200060
C                                                                         *S2200070
C      THIS PROGRAM SIMULATES THE EXECUTION OF A 2920 PROGRAM             *S2200080
C      FOR A FOURTH ORDER ELLIPTIC 600 HZ BANDPASS FILTER TRANSFER        *S2200090
C      TRANSFER FUNCTION WHICH IS THE PRODUCT OF TWO SECOND ORDER         *S2200100
C      FILTER SECTIONS                                                    *S2200110
C                                                                         *S2200120
C                        Y(Z)   N2(1,Z)   N2(2,Z)                         *S2200130
C                H(Z) = ---- = ------- X -------                          *S2200140
C                        X(Z)   D2(1,Z)   D2(2,Z)                         *S2200150
C                                                                         *S2200160
C      THE SIMULATION IS PERFORMED FOR AN IMPULSE INPUT EQUAL TO THE      *S2200170
C      GREATEST ALLOWABLE INPUT (SINMAG) AT T=0 AND ALLOWED TO RUN        *S2200180
C      OVER NUMIT ITERATIONS TO CHECK FOR STABILITY.                      *S2200190
C                                                                         *S2200200
C*************************************************************************S2200210
C VARIABLE DECLARATIONS                                                    S2200220
      INTEGER NUMIT,IMXAMP                                                 S2200230
      REAL X1(3),X2(3),Y1(3),Y2(3)                                         S2200240
      REAL SCALS1,SCALS2,SINMAG,MXAMP                                      S2200250
      REAL N2(2,3),D2(2,3),T,TX                                            S2200260
C DECLARE NUMBER OF SIMULATION ITERATIONS                                  S2200270
      NUMIT=8192                                                           S2200280
C PRINT OUTPUT HEADING                                                     S2200290
      WRITE(4,80)                                                          S2200300
   80 FORMAT(' S221 PROGRAM OUTPUT',//,                                    S2200310
     &' FOURTH ORDER FILTER IMPULSE RESPONSE',/,                           S2200320
     &'   (CASCADED SECOND ORDER SECTIONS)',///)                           S2200330
C FIRST SECOND ORDER STAGE COEFFICIENTS                                    S2200340
      N2(1,3)=1.                                                           S2200350
      N2(1,2)=-1.749875                                                    S2200360
      N2(1,1)=1.001585                                                     S2200370
      D2(1,3)=1.                                                           S2200380
      D2(1,2)=-1.81555                                                     S2200390
      D2(1,1)=0.990601                                                     S2200400
C PRINT FIRST STAGE TRANSFER FUNCTION                                      S2200410
      WRITE(4,81)N2(1,3),N2(1,2),N2(1,1),D2(1,3),D2(1,2),D2(1,1)           S2200420
   81 FORMAT(' FIRST STAGE TRANSFER FUNCTION (2920 COEFFICIENTS)',//,      S2200430
     &' ',E14.6,' + ',E14.6,' Z**-1 + ',E14.6,' Z**-2',/,                  S2200440
     &'   ----------------------------------------------------',/,S2200450
     &' ',E14.6,' + ',E14.6,' Z**-1 + ',E14.6,' Z**-2',///)                S2200460
C SECOND SECOND ORDER STAGE COEFFICIENTS                                   S2200470
      N2(2,3)=1.                                                           S2200480
      N2(2,2)=-1.877805                                                    S2200490
      N2(2,1)=0.998169                                                     S2200500
      D2(2,3)=1.                                                           S2200510
      D2(2,2)=-1.809446                                                    S2200520
      D2(2,1)=0.982554                                                     S2200530
C PRINT SECOND STAGE TRANSFER FUNCTION                                     S2200540
      WRITE(4,82)N2(2,3),N2(2,2),N2(2,1),D2(2,3),D2(2,2),D2(2,1)           S2200550
```

FILE: S221 FORTRAN A1

```
   82 FORMAT(' SECOND STAGE TRANSFER FUNCTION (2920 COEFFICIENTS)',//,  S2200560
     &' ',E14.6,' + ',E14.6,' Z**-1 + ',E14.6,' Z**-2',/,              S2200570
     &'   ----------------------------------------------------------',/,S2200580
     &' ',E14.6,' + ',E14.6,' Z**-1 + ',E14.6,' Z**-2',///)            S2200590
C INITIALIZE VARIABLES                                                 S2200600
      SCALS1=0.                                                        S2200610
      SCALS2=0.                                                        S2200620
C SAMPLE PERIOD = T = 1.1521152 X 10**-4 SECONDS                       S2200630
      T=4.*192./6.666E6                                                S2200640
C COMPUTE SCALE SUM FOR STAGES TO LIMIT INPUT AMPLITUDE                S2200650
      DO 6 K=1,3                                                       S2200660
         SCALS1=SCALS1+ABS(N2(1,K))                                    S2200670
         SCALS1=SCALS1+ABS(D2(1,K))                                    S2200680
         SCALS2=SCALS2+ABS(N2(2,K))                                    S2200690
         SCALS2=SCALS2+ABS(D2(2,K)).                                   S2200700
    6 CONTINUE                                                         S2200710
C ENSURE SCALE SUM FACTORS WILL LIMIT OUTPUT TO LESS THAN ONE          S2200720
      SCALS1=SCALS1+1.E-6                                              S2200730
      SCALS2=SCALS2+1.E-6                                              S2200740
C COMPUTE SIMULATED INPUT MAGNITUDE LIMIT                              S2200750
      SINMAG=1./(SCALS1*SCALS2)                                        S2200760
C PRINT STAGE SCALE SUMS AND INPUT MAGNITUDE LIMIT                     S2200770
      WRITE(4,85)SCALS1,SCALS2,SINMAG                                  S2200780
   85 FORMAT(//,' SCALE FACTORS AND INPUT MAGNITUDE LIMIT',///,        S2200790
     &' FIRST STAGE SCALE SUM = ',E14.6,/,                             S2200800
     &' SECOND STAGE SCALE SUM = ',E14.6,/,                            S2200810
     &' INPUT AMPLITUDE LIMITED TO +/- ',E14.6,////)                   S2200820
C PERFORM IMPULSE RESPONSE SIMULATION                                  S2200830
C INITIALIZE STAGE INPUTS AND OUTPUTS                                  S2200840
      DO 5 I=1,3                                                       S2200850
         X1(I)=0.                                                      S2200860
         X2(I)=0.                                                      S2200870
         Y1(I)=0.                                                      S2200880
         Y2(I)=0.                                                      S2200890
    5 CONTINUE                                                         S2200900
      MXAMP=0.                                                         S2200910
      IMXAMP=0                                                         S2200920
C PRINT SIMULATION HEADINGS                                            S2200930
      WRITE(4,98)                                                      S2200940
   98 FORMAT(' FILTER IMPULSE RESPONSE',//)                            S2200950
C     WRITE(4,99)                                                      S2200960
C  99 FORMAT(/,'     I      TIME             IN1              OUT2',//)  S2200970
C COMPUTE SIMULATED FILTER RESPONSE OVER REQUIRED ITERATIONS (NUMIT)    S2200980
      DO 10 I=1,NUMIT                                                  S2200990
C IN1 = X1(1) = IMPULSE INPUT AT T=0.                                  S2201000
         IF (I.EQ.1) X1(1)=SINMAG                                      S2201010
         IF (I.NE.1) X1(1)=0.                                          S2201020
C TX = TOTAL ELAPSED SAMPLE TIME                                       S2201030
         TX=T*FLOAT(I-1)                                               S2201040
C OUT1 = Y1(1) = FIRST STAGE OUTPUT = IN2 = X2(1) = SECOND STAGE INPUT S2201050
         Y1(1)=N2(1,3)*X1(1)+N2(1,2)*X1(2)+N2(1,1)*X1(3)-              S2201060
     &   D2(1,2)*Y1(2)-D2(1,1)*Y1(3)                                   S2201070
         X2(1)=Y1(1)                                                   S2201080
C OUT2 = Y2(1) = SECOND STAGE OUTPUT = XN(I)                           S2201090
         Y2(1)=N2(2,3)*X2(1)+N2(2,2)*X2(2)+N2(2,1)*X2(3)-              S2201100
```

```
     &   D2(2,2)*Y2(2)-D2(2,1)*Y2(3)                                    S2201110
         IF ((MXAMP.LT.(ABS(Y2(1)))).AND.(I.NE.1)) GO TO 30             S2201120
            GO TO 40                                                    S2201130
  30        MXAMP=ABS(Y2(1))                                           S2201140
            IMXAMP=I                                                    S2201150
  40        CONTINUE                                                   S2201160
C PRINT PARAMETERS FOR EACH SIMULATION ITERATION                        S2201170
C***********************************                                   S2201180
C         WRITE(4,100)IM1,TX,X1(1),Y2(1)                                S2201190
C 100     FORMAT(1X,I4,2X,3(F13.6,2X))                                  S2201200
C USE THIS OUTPUT FORMAT FOR EASYPLOT ROUTINE                           S2201210
C         WRITE(4,100)TX,X1(1),Y2(1)                                    S2201220
C 100     FORMAT(1X,3(F15.8,2X))                                        S2201230
C***********************************                                   S2201240
C PERFORM SIMULATION SHIFT DELAY                                        S2201250
         DO 15 J=2,3,1                                                  S2201260
            Y1(5-J)=Y1(4-J)                                             S2201270
            X1(5-J)=X1(4-J)                                             S2201280
            Y2(5-J)=Y2(4-J)                                             S2201290
            X2(5-J)=X2(4-J)                                             S2201300
  15     CONTINUE                                                       S2201310
         IF (MOD(I,512))10,14,10                                        S2201320
  14     WRITE(4,16)I,MXAMP                                             S2201330
  16     FORMAT(' MAX AMPLITUDE OF IMPULSE RESPONSE OVER ',I5,          S2201340
    &' ITERATIONS = ',F9.6)                                             S2201350
  10 CONTINUE                                                           S2201360
     WRITE(4,18)IMXAMP                                                  S2201370
  18 FORMAT(/,' MAX AMPLITUDE OCCURRED AT ITERATION ',I5)               S2201380
     STOP                                                               S2201390
     END                                                                S2201400
```

```
S221 PROGRAM OUTPUT

FOURTH ORDER FILTER IMPULSE RESPONSE
  (CASCADED SECOND ORDER SECTIONS)


FIRST STAGE TRANSFER FUNCTION (2920 COEFFICIENTS)

  0.100000E+01 +  -0.174988E+01 Z**-1 +   0.100159E+01 Z**-2
  ---------------------------------------------------------
  0.100000E+01 +  -0.181555E+01 Z**-1 +   0.990601E+00 Z**-2


SECOND STAGE TRANSFER FUNCTION (2920 COEFFICIENTS)

  0.100000E+01 +  -0.187780E+01 Z**-1 +   0.998169E+00 Z**-2
  ---------------------------------------------------------
  0.100000E+01 +  -0.180945E+01 Z**-1 +   0.982554E+00 Z**-2



SCALE FACTORS AND INPUT MAGNITUDE LIMIT

FIRST STAGE SCALE SUM =   0.755761E+01
SECOND STAGE SCALE SUM =   0.766797E+01
INPUT AMPLITUDE LIMITED TO +/-   0.172558E-01


FILTER IMPULSE RESPONSE

MAX AMPLITUDE OF IMPULSE RESPONSE OVER   512 ITERATIONS =  0.011935
MAX AMPLITUDE OF IMPULSE RESPONSE OVER  1024 ITERATIONS =  0.011935
MAX AMPLITUDE OF IMPULSE RESPONSE OVER  1536 ITERATIONS =  0.011935
MAX AMPLITUDE OF IMPULSE RESPONSE OVER  2048 ITERATIONS =  0.011935
MAX AMPLITUDE OF IMPULSE RESPONSE OVER  2560 ITERATIONS =  0.011935
MAX AMPLITUDE OF IMPULSE RESPONSE OVER  3072 ITERATIONS =  0.011935
MAX AMPLITUDE OF IMPULSE RESPONSE OVER  3584 ITERATIONS =  0.011935
MAX AMPLITUDE OF IMPULSE RESPONSE OVER  4096 ITERATIONS =  0.011935
MAX AMPLITUDE OF IMPULSE RESPONSE OVER  4608 ITERATIONS =  0.011935
MAX AMPLITUDE OF IMPULSE RESPONSE OVER  5120 ITERATIONS =  0.011935
MAX AMPLITUDE OF IMPULSE RESPONSE OVER  5632 ITERATIONS =  0.011935
MAX AMPLITUDE OF IMPULSE RESPONSE OVER  6144 ITERATIONS =  0.011935
MAX AMPLITUDE OF IMPULSE RESPONSE OVER  6656 ITERATIONS =  0.011935
MAX AMPLITUDE OF IMPULSE RESPONSE OVER  7168 ITERATIONS =  0.011935
MAX AMPLITUDE OF IMPULSE RESPONSE OVER  7680 ITERATIONS =  0.011935
MAX AMPLITUDE OF IMPULSE RESPONSE OVER  8192 ITERATIONS =  0.011935

MAX AMPLITUDE OCCURRED AT ITERATION   150
```

FILE: S221G FORTRAN A1

```
C*************************************************************************S2200010
C                                                                         *S2200020
C      THIS PROGRAM SIMULATES THE EXECUTION OF A 2920 PROGRAM             *S2200030
C      FOR A 600 HZ BANDPASS FILTER TRANSFER FUNCTION WHICH IS THE        *S2200040
C      PRODUCT OF TWO SECOND ORDER FILTER SECTIONS                        *S2200050
C                                                                         *S2200060
C               Y(Z)   DN213(Z)   DN223(Z)                                *S2200070
C        H(Z) = ---- = -------- X --------                                *S2200080
C               X(Z)   DD213(Z)   DD223(Z)                                *S2200090
C                                                                         *S2200100
C      THE SIMULATION IS PERFORMED FOR AN IMPULSE INPUT EQUAL TO THE      *S2200110
C      GREATEST ALLOWABLE INPUT (SINMAG) AT T=0.  WE THEN EXAMINE         *S2200120
C      THE FAST FOURIER TRANSFORM OF THE IMPULSE SEQUENCE OVER 1024       *S2200130
C      SAMPLES TO CONFIRM THE FREQUENCY RESPONSE OF THE SYSTEM.           *S2200140
C                                                                         *S2200150
C*************************************************************************S2200160
C VARIABLE DECLARATIONS                                                    S2200170
      INTEGER IM1,M,IWK(11)                                                S2200180
      REAL X1(3),X2(3),Y1(3),Y2(3)                                         S2200190
      REAL SCALS1,SCALS2,SINMAG                                            S2200200
      REAL DN213(3),DD213(3),DN223(3),DD223(3),T,TX                        S2200210
      COMPLEX A(1024)                                                      S2200220
      REAL XK(1024),XKHZ(1024),XN(1024),YMAG(1024),YPH(1024),Z(1024)       S2200230
C PRINT OUTPUT HEADING                                                     S2200240
      WRITE(4,80)                                                          S2200250
   80 FORMAT('EIGHTH ORDER FILTER IMPULSE RESPONSE',/,                     S2200260
     &'   (CASCADED SECOND ORDER SECTIONS)',/,                            S2200270
     &'(SECOND FOURTH ORDER BLOCK OF EIGHTH ORDER FILTER)',//)            S2200280
C FIRST SECOND ORDER STAGE COEFFICIENTS                                    S2200290
      DN213(3)=1.                                                          S2200300
      DN213(2)=-1.749875                                                   S2200310
      DN213(1)=1.001585                                                    S2200320
      DD213(3)=1.                                                          S2200330
      DD213(2)=-1.81555                                                    S2200340
      DD213(1)=0.990601                                                    S2200350
C PRINT FIRST STAGE TRANSFER FUNCTION                                      S2200360
      WRITE(4,81)DN213(3),DN213(2),DN213(1),DD213(3),DD213(2),DD213(1)     S2200370
   81 FORMAT('SECOND FIRST STAGE TRANSFER FUNCTION',//,                    S2200380
     &E14.6,' + ',E14.6,' Z**-1 + ',E14.6,' Z**-2',/,                      S2200390
     &' ------------------------------------------------------',/,S2200400
     &E14.6,' + ',E14.6,' Z**-1 + ',E14.6,' Z**-2',///)                    S2200410
C SECOND SECOND ORDER STAGE COEFFICIENTS                                   S2200420
      DN223(3)=1.                                                          S2200430
      DN223(2)=-1.877805                                                   S2200440
      DN223(1)=0.998169                                                    S2200450
      DD223(3)=1.                                                          S2200460
      DD223(2)=-1.809446                                                   S2200470
      DD223(1)=0.982544                                                    S2200480
C PRINT SECOND STAGE TRANSFER FUNCTION                                     S2200490
      WRITE(4,82)DN223(3),DN223(2),DN223(1),DD223(3),DD223(2),DD223(1)     S2200500
   82 FORMAT('SECOND SECOND STAGE TRANSFER FUNCTION',//,                   S2200510
     &E14.6,' + ',E14.6,' Z**-1 + ',E14.6,' Z**-2',/,                      S2200520
     &' ------------------------------------------------------',/,S2200530
     &E14.6,' + ',E14.6,' Z**-1 + ',E14.6,' Z**-2',///)                    S2200540
C INITIALIZE VARIABLES                                                     S2200550
```

FILE: S221G FORTRAN A1

```
      SCALS1=0.                                                          S2200560
      SCALS2=0.                                                          S2200570
C SAMPLE PERIOD = T = 1.1521152 X 10**-4 SECONDS                         S2200580
      T=4.*192./6.666E6                                                  S2200590
C COMPUTE SCALE SUM FOR STAGES TO LIMIT INPUT AMPLITUDE                  S2200600
      DO 6 K=1,3                                                         S2200610
         SCALS1=SCALS1+ABS(DN213(K))                                     S2200620
         SCALS1=SCALS1+ABS(DD213(K))                                     S2200630
         SCALS2=SCALS2+ABS(DN223(K))                                     S2200640
         SCALS2=SCALS2+ABS(DD223(K))                                     S2200650
    6 CONTINUE                                                           S2200660
C ENSURE SCALE SUM FACTORS WILL LIMIT OUTPUT TO LESS THAN ONE            S2200670
      SCALS1=SCALS1+1.E-6                                                S2200680
      SCALS2=SCALS2+1.E-6                                                S2200690
C COMPUTE SIMULATED INPUT MAGNITUDE LIMIT                                S2200700
      SINMAG=1./(SCALS1*SCALS2)                                          S2200710
C PRINT STAGE SCALE SUMS AND INPUT MAGNITUDE LIMIT                       S2200720
      WRITE(4,85)SCALS1,SCALS2,SINMAG                                    S2200730
   85 FORMAT(//,'SCALE FACTORS AND INPUT MAGNITUDE LIMIT',//,            S2200740
     &'FIRST STAGE SCALE SUM = ',E14.6,/,                                S2200750
     &'SECOND STAGE SCALE SUM = ',E14.6,/,                               S2200760
     &'INPUT AMPLITUDE LIMITED TO +/- ',E14.6,///)                       S2200770
C PERFORM IMPULSE RESPONSE SIMULATION                                    S2200780
C INITIALIZE STAGE INPUTS AND OUTPUTS                                    S2200790
      DO 5 I=1,3                                                         S2200800
         X1(I)=0.                                                        S2200810
         X2(I)=0.                                                        S2200820
         Y1(I)=0.                                                        S2200830
         Y2(I)=0.                                                        S2200840
    5 CONTINUE                                                           S2200850
C PRINT SIMULATION HEADINGS                                              S2200860
         WRITE(4,98)                                                     S2200870
   98    FORMAT(///,'FILTER IMPULSE RESPONSE',//)                        S2200880
         WRITE(4,99)                                                     S2200890
   99    FORMAT(/,'     I      TIME            IN1             OUT2',//)  S2200900
C COMPUTE SIMULATED FILTER RESPONSE OVER INDICATED NUMBER OF SAMPLES (I)S2200910
C*******************************                                       S2200920
C ADJUST AS NECESSARY                                                    S2200930
         DO 10 I=1,1024                                                  S2200940
C*******************************                                       S2200950
C IN1 = X1(1) = IMPULSE INPUT AT T=0.                                    S2200960
            IF (I.EQ.1) X1(1)=SINMAG                                     S2200970
            IF (I.NE.1) X1(1)=0.                                         S2200980
C TX = TOTAL ELAPSED SAMPLE TIME                                         S2200990
            TX=T*FLOAT(I-1)                                              S2201000
C OUT1 = Y1(1) = FIRST STAGE OUTPUT = IN2 = X2(1) = SECOND STAGE INPUT   S2201010
            Y1(1)=DN213(3)*X1(1)+DN213(2)*X1(2)+DN213(1)*X1(3)-          S2201020
     &      DD213(2)*Y1(2)-DD213(1)*Y1(3)                                S2201030
            X2(1)=Y1(1)                                                  S2201040
C OUT2 = Y2(1) = SECOND STAGE OUTPUT = FILTER OUTPUT                     S2201050
            Y2(1)=DN223(3)*X2(1)+DN223(2)*X2(2)+DN223(1)*X2(3)-          S2201060
     &      DD223(2)*Y2(2)-DD223(1)*Y2(3)                                S2201070
            XN(I)=Y2(1)                                                  S2201080
C FORM COMPLEX ARRAY OF IMPULSE VALUES FOR DFT                           S2201090
            A(I)=CMPLX(Y2(1),0.)                                         S2201100
```

FILE: S221G FORTRAN A1

```
C PRINT PARAMETERS FOR EACH SIMULATION ITERATION                          S2201110
C************************************                                    S2201120
            IM1=I-1                                                      S2201130
            WRITE(4,100)IM1,TX,X1(1),Y2(1)                               S2201140
  100       FORMAT(1X,I4,2X,3(F13.6,2X))                                 S2201150
C USE THIS OUTPUT FORMAT FOR EASYPLOT ROUTINE                            S2201160
C           WRITE(4,100)TX,X1(1),Y2(1)                                   S2201170
C 100       FORMAT(1X,3(F15.8,2X))                                       S2201180
C************************************                                    S2201190
C PERFORM SIMULATION SHIFT DELAY                                         S2201200
            DO 15 J=2,3,1                                                S2201210
               Y1(5-J)=Y1(4-J)                                           S2201220
               X1(5-J)=X1(4-J)                                           S2201230
               Y2(5-J)=Y2(4-J)                                           S2201240
               X2(5-J)=X2(4-J)                                           S2201250
   15       CONTINUE                                                     S2201260
   10    CONTINUE                                                        S2201270
   20 CONTINUE                                                           S2201280
C     THIS PROGRAM CALCULATES THE DFT OF THE IMPULSE RESPONSE            S2201290
C     OVER 1024 VALUES.  THIS IMPLIES M = 9 (2**9=1024).  IWK IS AN      S2201300
C     INTEGER VECTOR FOR FFT2C CALCULATION OF LENGTH M+1 = 10            S2201310
      M=10                                                               S2201320
      DO 30 I=1,1024                                                     S2201330
         A(I)=CONJG(A(I))                                                S2201340
   30 CONTINUE                                                           S2201350
      CALL FFT2C(A,M,IWK)                                                S2201360
      WRITE(4,38)                                                        S2201370
   38 FORMAT(///,1X,'DFT IMPULSE RESPONSE OUTPUT OVER 1024 ITERATIONS'   S2201380
     &,/,'       K            YMAG            YPHASE',/)                S2201390
      DO 40 I=1,1024                                                     S2201400
         Z(I-1)=0.                                                       S2201410
         A(I)=CONJG(A(I))                                                S2201420
         XK(I)=FLOAT(I-1)                                                S2201430
         XKHZ(I)=(1./T)*(XK(I)/1024.)                                    S2201440
         YMAG(I)=CABS(A(I))                                              S2201450
         YPH(I)=ATAN(AIMAG(A(I))/REAL(A(I)))                             S2201460
         WRITE(4,39)XK(I),YMAG(I),YPH(I)                                 S2201470
   39    FORMAT(1X,F8.0,4X,F12.4,4X,F12.4)                               S2201480
   40 CONTINUE                                                           S2201490
C--------------------------------------------------------------------    S2201500
C GRAPHICS PARAMETERS FOR YMAG VS K                                      S2201510
C--------------------------------------------------------------------    S2201520
      CALL LRGBUF                                                        S2201530
C     CALL TEK618                                                        S2201540
      CALL COMPRS                                                        S2201550
C .....     SETUP THE PLOTTING AREA                                      S2201560
      CALL PAGE (11.0,8.5)                                               S2201570
      CALL NOBRDR                                                        S2201580
      CALL AREA2D(9.0,6.5)                                               S2201590
C .....  LABEL THE X & Y AXES                                            S2201600
      CALL XNAME('FREQUENCY (HZ)$',100)                                  S2201610
      CALL YNAME('MAGNITUDE$',100)                                       S2201620
      CALL HEADIN ('DFT OF DIGITAL FILTER IMPULSE RESPONSE$',100,1.6,2)  S2201630
      CALL HEADIN ('MAGNITUDE VS FREQUENCY$',100,1.,2)                   S2201640
C .....   DEFINE THE AXES                                                S2201650
```

```
      CALL CROSS                                                        S2201660
      CALL GRAF(0.,100.,1100.,0.0,0.5,2.5)                              S2201670
C.....  DRAW THE BAR CURVES                                             S2201680
      CALL BARPAT(16)                                                   S2201690
      CALL BARWID(0.02)                                                 S2201700
      CALL VBARS(XKHZ,Z,YMAG,128)                                       S2201710
C.....  TERMINATE THIS PLOT                                            S2201720
      CALL ENDPL(0)                                                     S2201730
C------------------------------------------------------------------------ S2201740
C GRAPHICS PARAMETERS FOR YPHASE VS K                                   S2201750
C------------------------------------------------------------------------ S2201760
      CALL LRGBUF                                                       S2201770
C     CALL TEK618                                                       S2201780
      CALL COMPRS                                                       S2201790
C .....      SETUP THE PLOTTING AREA.                                   S2201800
      CALL PAGE (11.0,8.5)                                              S2201810
      CALL NOBRDR                                                       S2201820
      CALL AREA2D(9.0,6.5)                                              S2201830
C .....  LABEL THE X & Y AXES                                           S2201840
      CALL XNAME('FREQUENCY (HZ) $',100)                                S2201850
      CALL YNAME('PHASE (RADS)$',100)                                   S2201860
      CALL HEADIN ('DFT OF DIGITAL FILTER IMPULSE RESPONSE$',100,1.6,2) S2201870
      CALL HEADIN ('PHASE VS FREQUENCY$',100,1.,2)                      S2201880
C .....   DEFINE THE AXES                                              S2201890
      CALL CROSS                                                        S2201900
      CALL GRAF(0.,100.,1100.,-2.0,0.5,2.0)                             S2201910
C.....  DRAW THE PHASE CURVE                                           S2201920
      CALL THKCRV(0.01)                                                 S2201930
      CALL MARKER(15)                                                   S2201940
      CALL CURVE(XKHZ,YPH,128,0)                                        S2201950
C.....  TERMINATE THIS PLOT                                           S2201960
      CALL ENDPL(0)                                                     S2201970
C------------------------------------------------------------------------ S2201980
C GRAPHICS PARAMETERS FOR IMPULSE RESPONSE VS N                         S2201990
C------------------------------------------------------------------------ S2202000
      CALL LRGBUF                                                       S2202010
C     CALL TEK618                                                       S2202020
      CALL COMPRS                                                       S2202030
C .....      SETUP THE PLOTTING AREA                                    S2202040
      CALL PAGE (11.0,8.5)                                              S2202050
      CALL NOBRDR                                                       S2202060
      CALL AREA2D(9.0,6.5)                                              S2202070
C .....  LABEL THE X & Y AXES                                           S2202080
      CALL XNAME('ITERATION (N) $',100)                                 S2202090
      CALL YNAME('MAGNITUDE$',100)                                      S2202100
      CALL HEADIN ('DIGITAL FILTER IMPULSE RESPONSE$',100,1.6,3)        S2202110
      CALL HEADIN ('OUTPUT VS ITERATION $',100,1.,3)                    S2202120
      CALL HEADIN ('INPUT = .017 (SINMAG) AT T = 0$',100,1.,3)          S2202130
C .....   DEFINE THE AXES                                              S2202140
      CALL GRAF(0.,64.,1024.,-.03,.01,.03)                              S2202150
C.....  DRAW THE IMPULSE RESPONSE CURVE                                 S2202160
      CALL THKCRV(0.01)                                                 S2202170
      CALL MARKER(15)                                                   S2202180
      CALL CURVE(XK,XN,1024,0)                                          S2202190
C.....  TERMINATE THIS PLOT                                           S2202200
```

FILE: S221G FORTRAN A1

```
      CALL ENDPL(0)                                          S2202210
      CALL DONEPL                                            S2202220
      STOP                                                   S2202230
      END                                                    S2202240
```

FILE: S22F FORTRAN A1

```
C*************************************************************************S2200010
C                                                                         *S2200020
C                              APPENDIX G                                 *S2200030
C                                                                         *S2200040
C                         FORTRAN PROGRAM S22F                            *S2200050
C                                                                         *S2200060
C                                                                         *S2200070
C      THIS PROGRAM SIMULATES THE EXECUTION OF A 2920 PROGRAM             *S2200080
C      FOR A 600 HZ BANDPASS FILTER TRANSFER FUNCTION WHICH IS THE        *S2200090
C      PRODUCT OF TWO SECOND ORDER FILTER SECTIONS                        *S2200100
C                                                                         *S2200110
C               Y(Z)   DN213(Z)   DN223(Z)                                *S2200120
C        H(Z) = ---- = -------- X --------                                *S2200130
C               X(Z)   DD213(Z)   DD223(Z)                                *S2200140
C                                                                         *S2200150
C      THE SIMULATION IS PERFORMED OVER A RANGE OF DIFFERENT INPUT        *S2200160
C      FREQUENCIES ABOUT THE TARGET CENTER FREQUENCY OF 600 HZ            *S2200170
C                                                                         *S2200180
C*************************************************************************S2200190
C VARIABLE DECLARATIONS                                                    S2200200
      INTEGER IMOUT                                                        S2200210
      REAL TX,IN0,OUT0,F(9)                                                S2200220
      REAL X1(3),X2(3),Y1(3),Y2(3)                                         S2200230
      REAL SCALS1,SCALS2,SINMAG,MOUT                                       S2200240
      REAL DN213(3),DD213(3),DN223(3),DD223(3),T,PI,TWOPIF                 S2200250
C INPUT FREQUENCIES                                                        S2200260
      F(1)=500.                                                            S2200270
      F(2)=575.                                                            S2200280
      F(3)=590.825                                                         S2200290
      F(4)=600.                                                            S2200300
      F(5)=625.                                                            S2200310
      F(6)=700.                                                            S2200320
C     F(7)=                                                                S2200330
C     F(8)=                                                                S2200340
C     F(9)=                                                                S2200350
      WRITE(4,80)                                                          S2200360
   80 FORMAT(' PROGRAM S22F OUTPUT',//,                                    S2200370
     &' FOURTH ORDER FILTER FREQUENCY RESPONSE',/,                         S2200380
     &'   (CASCADED SECOND ORDER SECTIONS)',///)                          S2200390
C FIRST SECOND ORDER STAGE COEFFICIENTS                                    S2200400
      DN213(3)=1.                                                          S2200410
      DN213(2)=-1.749875                                                   S2200420
      DN213(1)=1.001585                                                    S2200430
      DD213(3)=1.                                                          S2200440
      DD213(2)=-1.81555                                                    S2200450
      DD213(1)=0.990601                                                    S2200460
C PRINT FIRST STAGE TRANSFER FUNCTION                                      S2200470
      WRITE(4,81)DN213(3),DN213(2),DN213(1),DD213(3),DD213(2),DD213(1)     S2200480
   81 FORMAT(' FIRST STAGE TRANSFER FUNCTION (2920 EQUIVALENT)',//,        S2200490
     &' ',E14.6,' + ',E14.6,' Z**-1 + ',E14.6,' Z**-2',/,                  S2200500
     &'   ------------------------------------------------------',/,       S2200510
     &' ',E14.6,' + ',E14.6,' Z**-1 + ',E14.6,' Z**-2',///)                S2200520
C SECOND SECOND ORDER STAGE COEFFICIENTS                                   S2200530
      DN223(3)=1.                                                          S2200540
      DN223(2)=-1.877805                                                   S2200550
```

```
      DN223(1)=0.998169                                                 S2200560
      DD223(3)=1.                                                       S2200570
      DD223(2)=-1.809446                                                S2200580
      DD223(1)=0.982544                                                 S2200590
C PRINT SECOND STAGE TRANSFER FUNCTION                                  S2200600
      WRITE(4,82)DN223(3),DN223(2),DN223(1),DD223(3),DD223(2),DD223(1)  S2200610
   82 FORMAT(' SECOND STAGE TRANSFER FUNCTION (2920 EQUIVALENT)',//,    S2200620
     &' ',E14.6,' + ',E14.6,' Z**-1 + ',E14.6,' Z**-2',/,               S2200630
     &'   ----------------------------------------------------------',/,S2200640
     &' ',E14.6,' + ',E14.6,' Z**-1 + ',E14.6,' Z**-2',///)             S2200650
C INITIALIZE VARIABLES                                                  S2200660
      PI=3.1415927                                                      S2200670
      SCALS1=0.                                                         S2200680
      SCALS2=0.                                                         S2200690
C SAMPLE PERIOD = T = 1.1521152 X 10**-4 SECONDS                        S2200700
      T=4.*192./6.666E6                                                 S2200710
C COMPUTE SCALE SUM FOR STAGES TO LIMIT INPUT AMPLITUDE                 S2200720
      DO 6 K=1,3                                                        S2200730
         SCALS1=SCALS1+ABS(DN213(K))                                    S2200740
         SCALS1=SCALS1+ABS(DD213(K))                                    S2200750
         SCALS2=SCALS2+ABS(DN223(K))                                    S2200760
         SCALS2=SCALS2+ABS(DD223(K))                                    S2200770
    6 CONTINUE                                                          S2200780
C ENSURE SCALE.SUM FACTORS WILL LIMIT OUTPUT TO LESS THAN ONE           S2200790
      SCALS1=SCALS1+1.E-6                                               S2200800
      SCALS2=SCALS2+1.E-6                                               S2200810
C COMPUTE SIMULATED INPUT MAGNITUDE LIMIT                               S2200820
      SINMAG=1./(SCALS1*SCALS2)                                         S2200830
C PRINT STAGE SCALE SUMS AND INPUT MAGNITUDE LIMIT                      S2200840
      WRITE(4,85)SCALS1,SCALS2,SINMAG                                   S2200850
   85 FORMAT(//,' SCALE FACTORS AND INPUT MAGNITUDE LIMIT',//,          S2200860
     &' FIRST STAGE SCALE SUM = ',E14.6,/,                              S2200870
     &' SECOND STAGE SCALE SUM = ',E14.6,/,                             S2200880
     &' INPUT AMPLITUDE LIMITED TO +/- ',E14.6,///)                     S2200890
      WRITE(4,86)                                                       S2200900
   86 FORMAT('1')                                                       S2200910
C BEGIN SIMULATION FOR SPECIFIED FREQUENCIES GIVEN BY F(L)               S2200920
C*********************************                                     S2200930
C ADJUST AS NECESSARY                                                   S2200940
      DO 20 L=1,6                                                       S2200950
C*********************************                                     S2200960
C COMPUTE SIMULATION RUN INPUT CONSTANT FOR EACH FREQUENCY              S2200970
         TWOPIF=2.*PI*F(L)                                              S2200980
C INITIALIZE STAGE INPUTS AND OUTPUTS                                   S2200990
         IMOUT=0                                                        S2201000
         MOUT=0.                                                        S2201010
         DO 5 I=1,3                                                     S2201020
            X1(I)=0.                                                    S2201030
            X2(I)=0.                                                    S2201040
            Y1(I)=0.                                                    S2201050
            Y2(I)=0.                                                    S2201060
    5    CONTINUE                                                       S2201070
C PRINT SIMULATION HEADINGS                                             S2201080
         WRITE(4,98)F(L)                                                S2201090
   98    FORMAT(' FILTER FREQUENCY RESPONSE FOR F = ',F5.0,' HZ',/)     S2201100
```

FILE: S22F FORTRAN A1

```
C        WRITE(4,99)                                                    S2201110
C  99    FORMAT(/,'     I      TIME        IN1      OUT1=IN2     OUT2',//) S2201120
C COMPUTE SIMULATED FILTER RESPONSE OVER INDICATED NUMBER OF SAMPLES (I)S2201130
C*******************************                                       S2201140
C ADJUST AS NECESSARY                                                   S2201150
         DO 10 I=1,2048                                                 S2201160
C*******************************                                       S2201170
C TX = TOTAL ELAPSED SAMPLE TIME                                        S2201180
            TX=T*FLOAT(I-1)                                             S2201190
C IN1 = X1(1) = FILTER FIRST STAGE INPUT VALUE (LIMITED BY SINMAG)      S2201200
            X1(1)=SINMAG*SIN(TWOPIF*TX)                                 S2201210
C OUT1 = Y1(1) = FIRST STAGE OUTPUT = IN2 = X2(1) = SECOND STAGE INPUT  S2201220
            Y1(1)=DN213(3)*X1(1)+DN213(2)*X1(2)+DN213(1)*X1(3)-         S2201230
     &      DD213(2)*Y1(2)-DD213(1)*Y1(3)                               S2201240
            X2(1)=Y1(1)                                                 S2201250
C OUT2 = Y2(1) = SECOND STAGE OUTPUT = FILTER OUTPUT                    S2201260
            Y2(1)=DN223(3)*X2(1)+DN223(2)*X2(2)+DN223(1)*X2(3)-         S2201270
   &      DD223(2)*Y2(2)-DD223(1)*Y2(3)                                S2201280
C PRINT PARAMETERS FOR EACH SIMULATION ITERATION                        S2201290
C***********************************                                  S2201300
C           WRITE(4,100)I,TX,X1(1),X2(1),Y2(1)                          S2201310
C 100       FORMAT(1X,I4,2X,4(F10.3,2X))                                S2201320
C USE THIS OUTPUT FORMAT FOR EASYPLOT ROUTINE                           S2201330
C           WRITE(4,100)TX,X1(1),Y2(1)                                  S2201340
C 100       FORMAT(1X,3(F15.8,2X))                                      S2201350
C***********************************                                  S2201360
C REMEMBER MAXIMUM AMPLITUDE IN EACH FREQUENCY SIMULATION TRIAL         S2201370
            IF (ABS(Y2(1))-MOUT) 11,11,14                               S2201380
   14          MOUT=ABS(Y2(1))                                          S2201390
               IMOUT=I                                                  S2201400
C PERFORM SIMULATION SHIFT DELAY                                        S2201410
   11       DO 15 J=2,3,1                                              S2201420
               Y1(5-J)=Y1(4-J)                                          S2201430
               X1(5-J)=X1(4-J)                                          S2201440
               Y2(5-J)=Y2(4-J)                                          S2201450
               X2(5-J)=X2(4-J)                                          S2201460
   15       CONTINUE                                                   S2201470
   10    CONTINUE                                                      S2201480
C PRINT MAXIMUM OUTPUT AMPLITUDE FOR EACH FREQUENCY SIMULATION RUN      S2201490
         WRITE(4,89)MOUT,IMOUT                                          S2201500
   89    FORMAT('   MAXIMUM OUTPUT AMPLITUDE = ',F15.8,/,               S2201510
     &'   THIS OCCURRED AT SIMULATION ITERATION ',I5,///)               S2201520
   20 CONTINUE                                                          S2201530
      STOP                                                              S2201540
      END                                                               S2201550
```

```
PROGRAM S22F OUTPUT

FOURTH ORDER FILTER FREQUENCY RESPONSE
  (CASCADED SECOND ORDER SECTIONS)


FIRST STAGE TRANSFER FUNCTION (2920 EQUIVALENT)

  0.100000E+01 +  -0.174988E+01 Z**-1 +   0.100159E+01 Z**-2
  ---------------------------------------------------------
  0.100000E+01 +  -0.181555E+01 Z**-1 +   0.990601E+00 Z**-2


SECOND STAGE TRANSFER FUNCTION (2920 EQUIVALENT)

  0.100000E+01 +  -0.187780E+01 Z**-1 +   0.998169E+00 Z**-2
  ---------------------------------------------------------
  0.100000E+01 +  -0.180945E+01 Z**-1 +   0.982544E+00 Z**-2



SCALE FACTORS AND INPUT MAGNITUDE LIMIT

FIRST STAGE SCALE SUM =   0.755761E+01
SECOND STAGE SCALE SUM =   0.766796E+01
INPUT AMPLITUDE LIMITED TO +/-   0.172558E-01
```

```
FILTER FREQUENCY RESPONSE FOR F =  500. HZ

  MAXIMUM OUTPUT AMPLITUDE =      0.10085225
  THIS OCCURRED AT SIMULATION ITERATION   180


FILTER FREQUENCY RESPONSE FOR F =  575. HZ

  MAXIMUM OUTPUT AMPLITUDE =      1.59343243
  THIS OCCURRED AT SIMULATION ITERATION   515


FILTER FREQUENCY RESPONSE FOR F =  591. HZ

  MAXIMUM OUTPUT AMPLITUDE =      1.62147427
  THIS OCCURRED AT SIMULATION ITERATION   500


FILTER FREQUENCY RESPONSE FOR F =  600. HZ

  MAXIMUM OUTPUT AMPLITUDE =      0.89969766
  THIS OCCURRED AT SIMULATION ITERATION   284


FILTER FREQUENCY RESPONSE FOR F =  625. HZ

  MAXIMUM OUTPUT AMPLITUDE =      0.30743510
  THIS OCCURRED AT SIMULATION ITERATION   135


FILTER FREQUENCY RESPONSE FOR F =  700. HZ

  MAXIMUM OUTPUT AMPLITUDE =      0.07697707
  THIS OCCURRED AT SIMULATION ITERATION   150
```

FILE: S22FG FORTRAN A1

```
C*************************************************************************S2200010
C                                                                         *S2200020
C      THIS PROGRAM SIMULATES THE EXECUTION OF A 2920 PROGRAM             *S2200030
C      FOR A 600 HZ BANDPASS FILTER TRANSFER FUNCTION WHICH IS THE        *S2200040
C      PRODUCT OF TWO SECOND ORDER FILTER SECTIONS                        *S2200050
C                                                                         *S2200060
C               Y(Z)   DN213(Z)   DN223(Z)                                *S2200070
C        H(Z) = ---- = -------- X --------                                *S2200080
C               X(Z)   DD213(Z)   DD223(Z)                                *S2200090
C                                                                         *S2200100
C      THE SIMULATION IS PERFORMED OVER A RANGE OF DIFFERENT INPUT        *S2200110
C      FREQUENCIES ABOUT THE TARGET CENTER FREQUENCY OF 600 HZ.           *S2200120
C      AFTER SIMULATION THE FREQUENCY RESPONSE IS PLOTTED FOR             *S2200130
C      GRAPHICAL REVIEW.                                                  *S2200140
C                                                                         *S2200150
C*************************************************************************S2200160
C VARIABLE DECLARATIONS                                                    S2200170
      INTEGER IMOUT                                                        S2200180
      REAL TX,F(9)                                                         S2200190
      REAL NUM(1024)                                                       S2200200
      REAL IN(1024)                                                        S2200210
      REAL OUT(1024)                                                       S2200220
      REAL X1(3),X2(3),Y1(3),Y2(3)                                         S2200230
      REAL SCALS1,SCALS2,SINMAG,MOUT                                       S2200240
      REAL DN213(3),DD213(3),DN223(3),DD223(3),T,PI,TWOPIF                 S2200250
C INPUT FREQUENCIES                                                        S2200260
      F(1)=700                                                             S2200270
C     F(2)=                                                                S2200280
C     F(3)=                                                                S2200290
C     F(4)=                                                                S2200300
C     F(5)=                                                                S2200310
C     F(6)=                                                                S2200320
C     F(7)=                                                                S2200330
C     F(8)=                                                                S2200340
C     F(9)=                                                                S2200350
      WRITE(4,80)                                                          S2200360
   80 FORMAT('FOURTH ORDER FILTER FREQUENCY RESPONSE',/,                   S2200370
     &'   (CASCADED SECOND ORDER SECTIONS)',//)                           S2200380
C FIRST SECOND ORDER STAGE COEFFICIENTS                                    S2200390
      DN213(3)=1.                                                          S2200400
      DN213(2)=-1.749875                                                   S2200410
      DN213(1)=1.001585                                                    S2200420
      DD213(3)=1.                                                          S2200430
      DD213(2)=-1.81555                                                    S2200440
      DD213(1)=0.990601                                                    S2200450
C PRINT FIRST STAGE TRANSFER FUNCTION                                      S2200460
      WRITE(4,81)DN213(3),DN213(2),DN213(1),DD213(3),DD213(2),DD213(1)     S2200470
   81 FORMAT('FIRST STAGE TRANSFER FUNCTION (2920 EQUIVALENT)',//,         S2200480
     &E14.6,' + ',E14.6,' Z**-1 + ',E14.6,' Z**-2',/,                      S2200490
     &' ------------------------------------------------------',/,S2200500
     &E14.6,' + ',E14.6,' Z**-1 + ',E14.6,' Z**-2',///)                    S2200510
C SECOND SECOND ORDER STAGE COEFFICIENTS                                   S2200520
      DN223(3)=1.                                                          S2200530
      DN223(2)=-1.877805                                                   S2200540
      DN223(1)=0,998169                                                    S2200550
```

FILE: S22FG FORTRAN A1

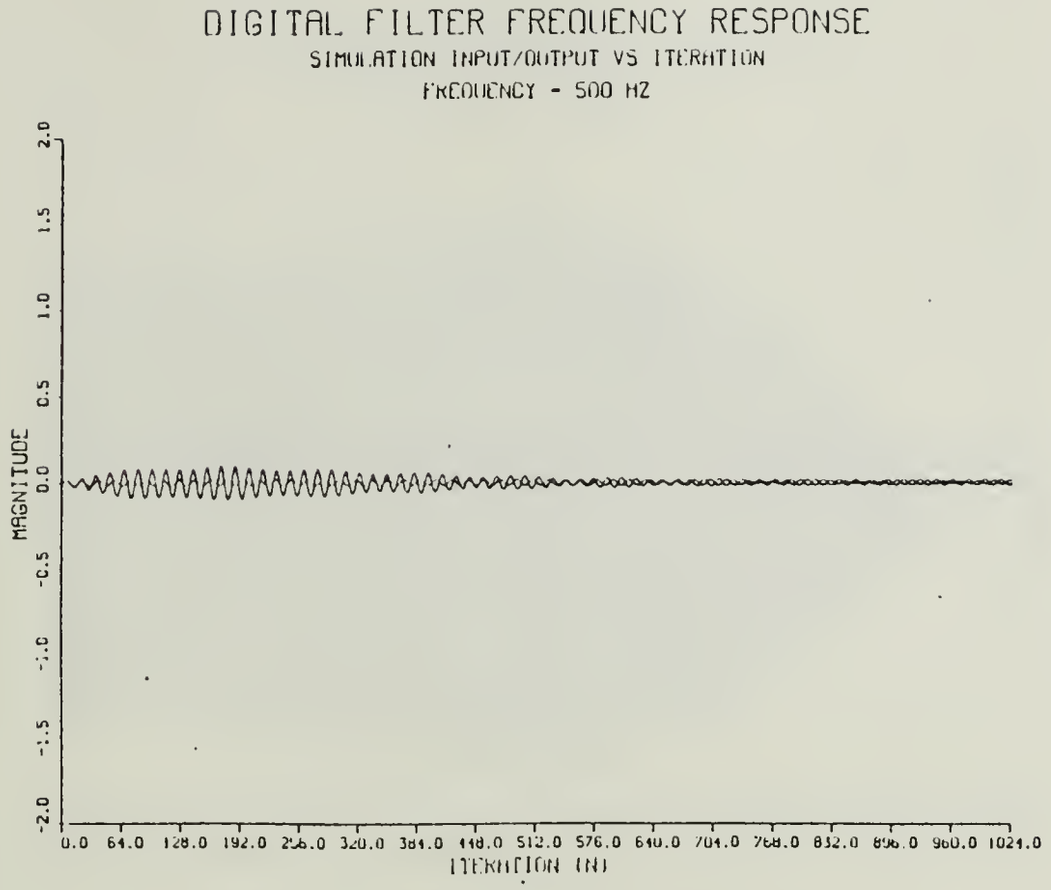
```
      DD223(3)=1.                                                        S2200560
      DD223(2)=-1.809446                                                 S2200570
      DD223(1)=0.982544                                                  S2200580
C PRINT SECOND STAGE TRANSFER FUNCTION                                   S2200590
      WRITE(4,82)DN223(3),DN223(2),DN223(1),DD223(3),DD223(2),DD223(1)   S2200600
   82 FORMAT('SECOND STAGE TRANSFER FUNCTION (2920 EQUIVALENT)',//,      S2200610
     &E14.6,' + ',E14.6,' Z**-1 + ',E14.6,' Z**-2',/,                    S2200620
     &' ------------------------------------------------------------',/,S2200630
     &E14.6,' + ',E14.6,' Z**-1 + ',E14.6,' Z**-2',///)                  S2200640
C INITIALIZE VARIABLES                                                   S2200650
      PI=3.1415927                                                       S2200660
      SCALS1=0.                                                          S2200670
      SCALS2=0.                                                          S2200680
C SAMPLE PERIOD = T = 1.1521152 X 10**-4 SECONDS                         S2200690
      T=4.*192./6.666E6                                                  S2200700
C COMPUTE SCALE SUM FOR STAGES TO LIMIT INPUT AMPLITUDE                  S2200710
      DO 6 K=1,3                                                         S2200720
         SCALS1=SCALS1+ABS(DN213(K))                                     S2200730
         SCALS1=SCALS1+ABS(DD213(K))                                     S2200740
         SCALS2=SCALS2+ABS(DN223(K))                                     S2200750
         SCALS2=SCALS2+ABS(DD223(K))                                     S2200760
    6 CONTINUE                                                           S2200770
C ENSURE SCALE SUM FACTORS WILL LIMIT OUTPUT TO LESS THAN ONE            S2200780
      SCALS1=SCALS1+1.E-6                                                S2200790
      SCALS2=SCALS2+1.E-6                                                S2200800
C COMPUTE SIMULATED INPUT MAGNITUDE LIMIT                                S2200810
      SINMAG=1./(SCALS1*SCALS2)                                          S2200820
C PRINT STAGE SCALE SUMS AND INPUT MAGNITUDE LIMIT                       S2200830
      WRITE(4,85)SCALS1,SCALS2,SINMAG                                    S2200840
   85 FORMAT(//,'SCALE FACTORS AND INPUT MAGNITUDE LIMIT',//,            S2200850
     &'FIRST STAGE SCALE SUM = ',E14.6,/,                                S2200860
     &'SECOND STAGE SCALE SUM = ',E14.6,/,                               S2200870
     &'INPUT AMPLITUDE LIMITED TO +/- ',E14.6,///)                       S2200880
C BEGIN SIMULATION FOR SPECIFIED FREQUENCIES GIVEN BY F(L)                S2200890
C*******************************                                        S2200900
C ADJUST AS NECESSARY                                                    S2200910
      DO 20 L=1,1                                                        S2200920
C*******************************                                        S2200930
C COMPUTE SIMULATION RUN INPUT CONSTANT FOR EACH FREQUENCY                S2200940
      TWOPIF=2.*PI*F(L)                                                  S2200950
C INITIALIZE STAGE INPUTS AND OUTPUTS                                    S2200960
      IMOUT=0                                                            S2200970
      MOUT=0.                                                            S2200980
      DO 5 I=1,3                                                         S2200990
         X1(I)=0.                                                        S2201000
         X2(I)=0.                                                        S2201010
         Y1(I)=0.                                                        S2201020
         Y2(I)=0.                                                        S2201030
    5 CONTINUE                                                           S2201040
C PRINT SIMULATION HEADINGS                                              S2201050
      WRITE(4,98)F(L)                                                    S2201060
   98 FORMAT(///,'FILTER FREQUENCY RESPONSE FOR F = ',F5.0,' HZ',//)     S2201070
      WRITE(4,99)                                                        S2201080
   99 FORMAT(/,'     I      TIME        IN1      OUT1=IN2     OUT2',//)  S2201090
C COMPUTE SIMULATED FILTER RESPONSE OVER INDICATED NUMBER OF SAMPLES (I)S2201100
```

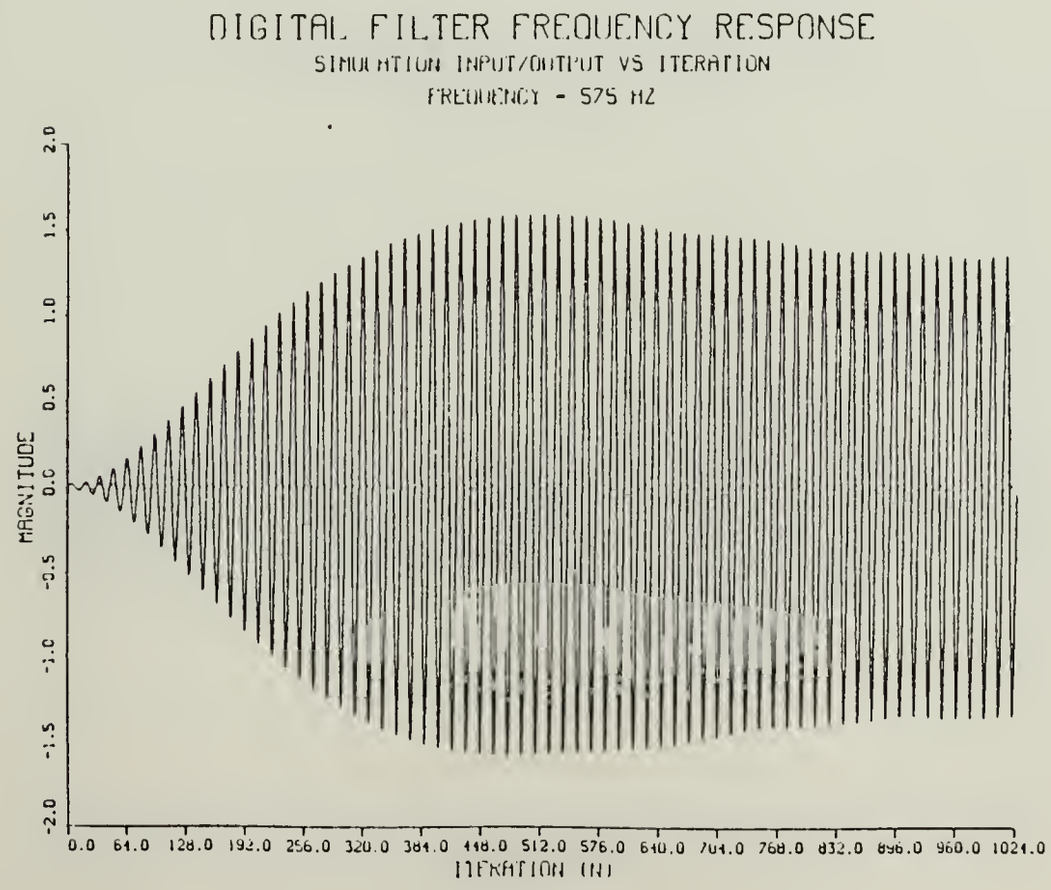
FILE: S22FG FORTRAN A1

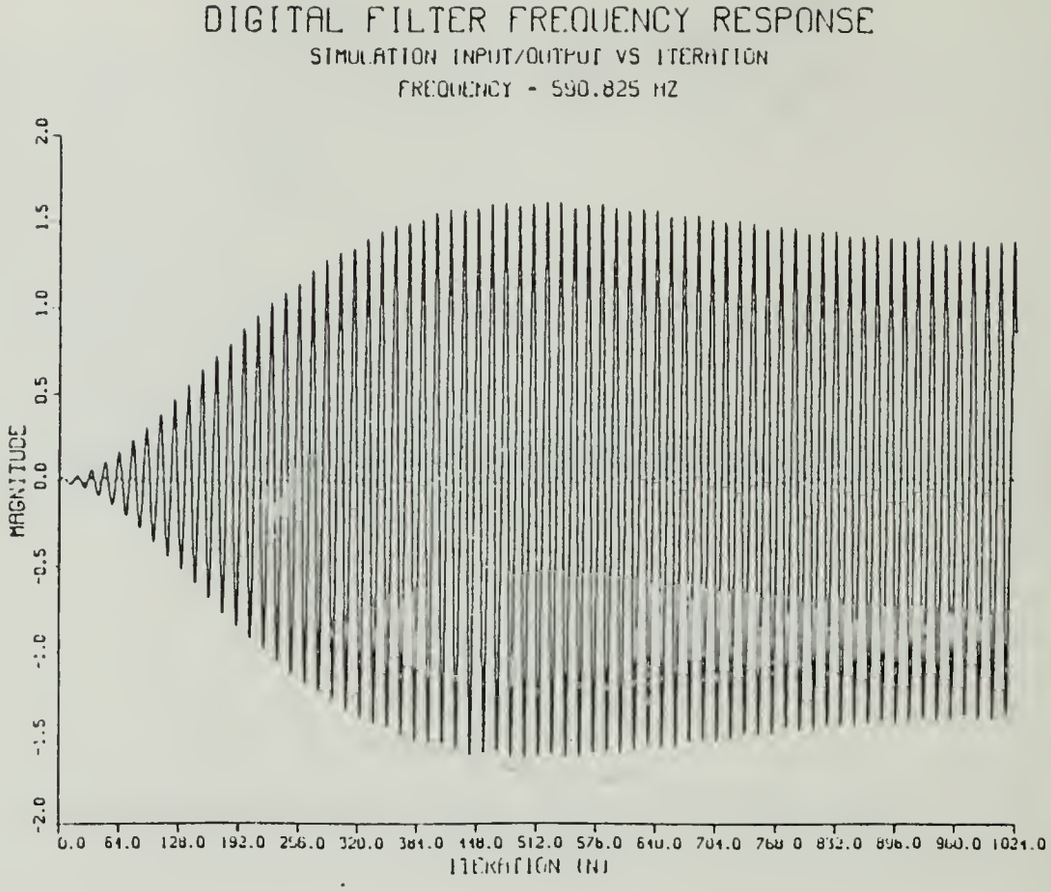
```
C*********************************                                    S2201110
C ADJUST AS NECESSARY                                                 S2201120
         DO 10 I=1,1024                                               S2201130
C*********************************                                    S2201140
C TX = TOTAL ELAPSED SAMPLE TIME                                      S2201150
            TX=T*FLOAT(I-1)                                           S2201160
C IN1 = X1(1) = FILTER FIRST STAGE INPUT VALUE (LIMITED BY SINMAG)    S2201170
            X1(1)=SINMAG*SIN(TWOPIF*TX)                               S2201180
            NUM(I)=FLOAT(I)                                           S2201190
            IN(I)=X1(1)                                               S2201200
C OUT1 = Y1(1) = FIRST STAGE OUTPUT = IN2 = X2(1) = SECOND STAGE INPUT S2201210
            Y1(1)=DN213(3)*X1(1)+DN213(2)*X1(2)+DN213(1)*X1(3)-       S2201220
     &      DD213(2)*Y1(2)-DD213(1)*Y1(3)                             S2201230
            X2(1)=Y1(1)                                               S2201240
C OUT2 = Y2(1) = SECOND STAGE OUTPUT = FILTER OUTPUT                  S2201250
            Y2(1)=DN223(3)*X2(1)+DN223(2)*X2(2)+DN223(1)*X2(3)-       S2201260
     &      DD223(2)*Y2(2)-DD223(1)*Y2(3)                             S2201270
            OUT(I)=Y2(1)                                              S2201280
C PRINT PARAMETERS FOR EACH SIMULATION ITERATION                      S2201290
C*********************************                                    S2201300
C           WRITE(4,100)I,TX,X1(1),X2(1),Y2(1)                        S2201310
C 100       FORMAT(1X,I4,2X,4(F10.3,2X))                              S2201320
C USE THIS OUTPUT FORMAT FOR EASYPLOT ROUTINE                         S2201330
C           WRITE(4,100)TX,X1(1),Y2(1)                                S2201340
C 100       FORMAT(1X,3(F15.8,2X))                                    S2201350
C*********************************                                    S2201360
C REMEMBER MAXIMUM AMPLITUDE IN EACH FREQUENCY SIMULATION TRIAL       S2201370
            IF (ABS(Y2(1))-MOUT) 11,11,14                             S2201380
   14          MOUT=ABS(Y2(1))                                        S2201390
               IMOUT=I                                                S2201400
C PERFORM SIMULATION SHIFT DELAY                                      S2201410
   11       DO 15 J=2,3,1                                             S2201420
               Y1(5-J)=Y1(4-J)                                        S2201430
               X1(5-J)=X1(4-J)                                        S2201440
               Y2(5-J)=Y2(4-J)                                        S2201450
               X2(5-J)=X2(4-J)                                        S2201460
   15          CONTINUE                                               S2201470
   10     CONTINUE                                                    S2201480
C PRINT MAXIMUM OUTPUT AMPLITUDE FOR EACH FREQUENCY SIMULATION RUN    S2201490
         WRITE(4,89)F(L),MOUT,IMOUT                                   S2201500
   89     FORMAT(' MAXIMUM OUTPUT AMPLITUDE FOR ',F5.0,' HZ = ',F15.8,/, S2201510
     &' THIS OCCURRED AT SIMULATION ITERATION ',I5)                   S2201520
   20 CONTINUE                                                        S2201530
C------------------------------------------------------------------- S2201540
C GRAPHICS PARAMETERS FOR FREQUENCY RESPONSE OUTPUT VS INPUT          S2201550
C------------------------------------------------------------------- S2201560
      CALL LRGBUF                                                     S2201570
C     CALL TEK618                                                     S2201580
      CALL COMPRS                                                     S2201590
C .....     SETUP THE PLOTTING AREA                                   S2201600
      CALL PAGE (11.0,8.5)                                            S2201610
      CALL NOBRDR                                                     S2201620
      CALL AREA2D(9.0,6.5)                                            S2201630
C .....  LABEL THE X & Y AXES                                         S2201640
      CALL XNAME('ITERATION (N) $',100)                               S2201650
```

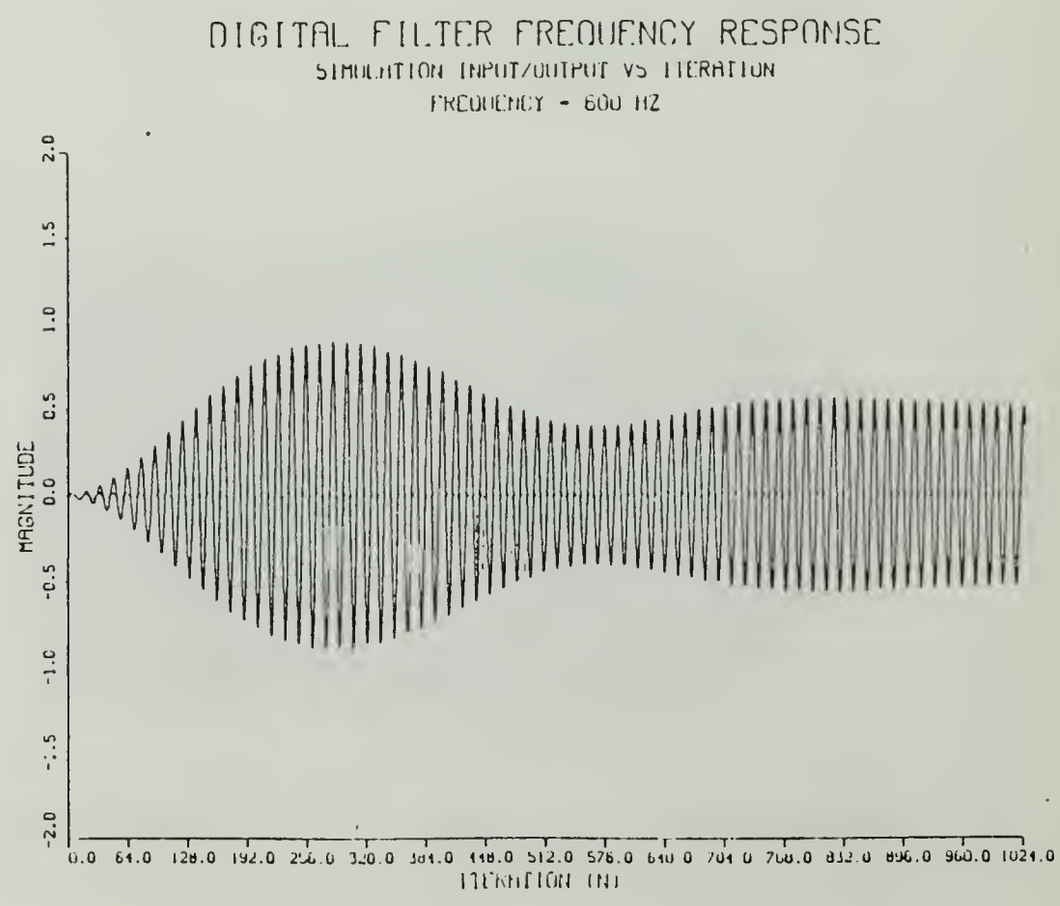
FILE: S22FG FORTRAN A1

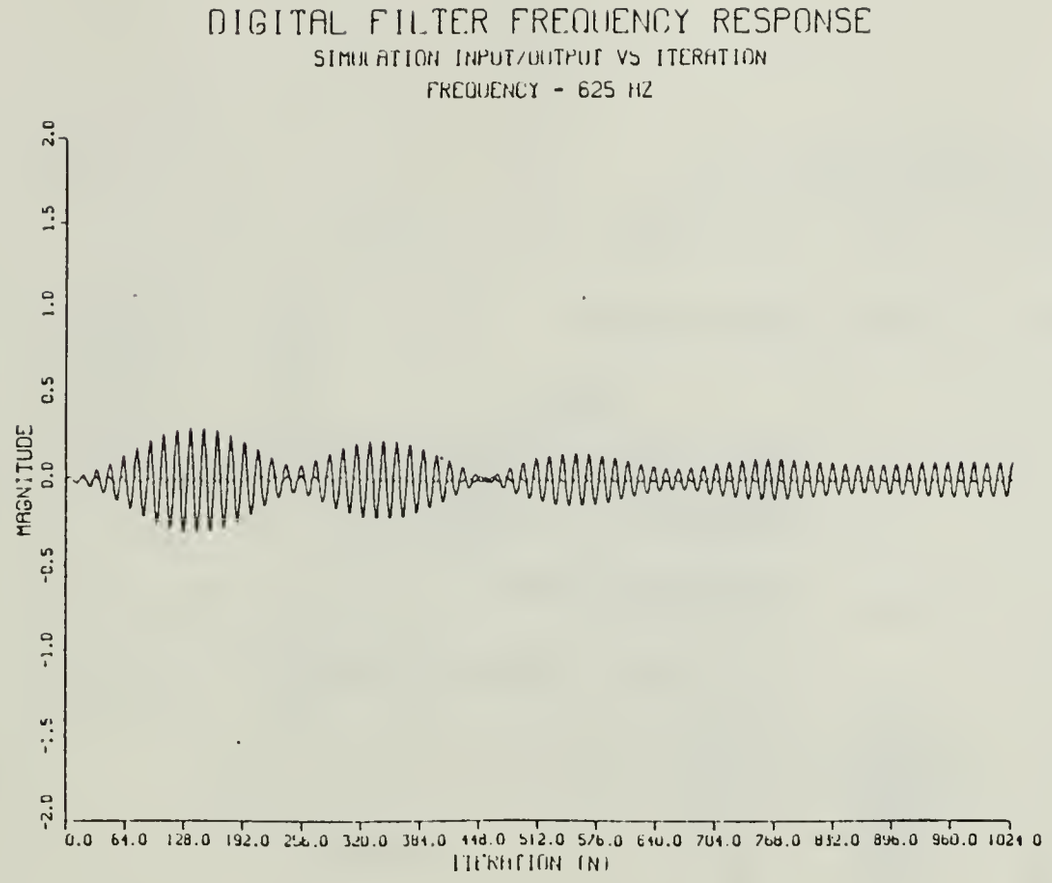
```
      CALL YNAME('MAGNITUDES$',100)                                    S2201660
      CALL HEADIN ('DIGITAL FILTER FREQUENCY RESPONSES$',100,1.6,3)    S2201670
      CALL HEADIN ('SIMULATION INPUT/OUTPUT VS ITERATION$',100,1.,3)   S2201680
      CALL HEADIN ('FREQUENCY = 700 HZ$',100,1.,3)                     S2201690
C .....   DEFINE THE AXES                                             S2201700
      CALL GRAF(0.,64.,1024.,-2.0,.5,2.0)                              S2201710
C..... DRAW THE INPUT CURVE                                            S2201720
      CALL THKCRV(0.01)                                                S2201730
      CALL MARKER(15)                                                  S2201740
      CALL CURVE(NUM,IN,1024,0)                                        S2201750
C..... DRAW THE OUTPUT CURVE                                           S2201760
      CALL THKCRV(0.01)                                                S2201770
      CALL MARKER(15)                                                  S2201780
      CALL CURVE(NUM,OUT,1024,0)                                       S2201790
C..... TERMINATE THIS PLOT                                             S2201800
      CALL ENDPL(0)                                                    S2201810
      CALL DONEPL                                                      S2201820
      STOP                                                             S2201830
      END                                                              S2201840
```

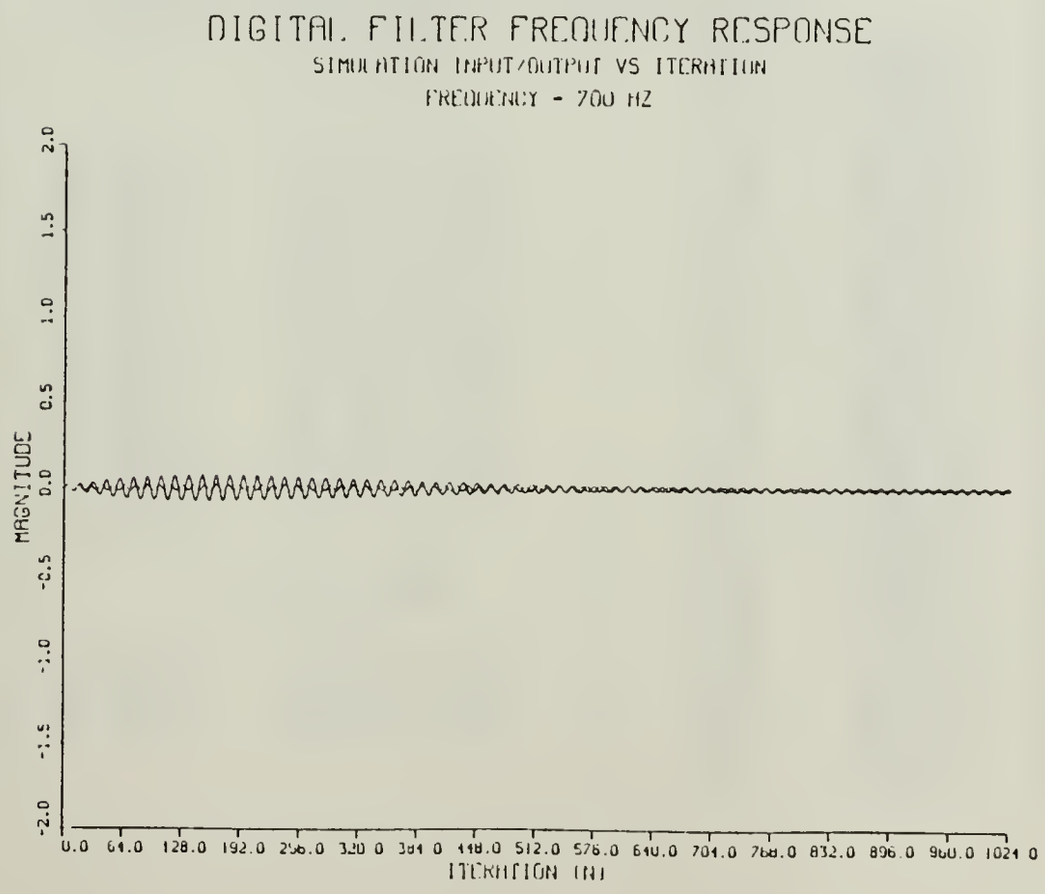

DIGITAL FILTER FREQUENCY RESPONSE
SIMULATION INPUT/OUTPUT VS ITERATION
FREQUENCY - 500 HZ
MAGNITUDE
2.0
1.5
1.0
0.5
0.0
-0.5
-1.0
-1.5
-2.0
0.0 64.0 128.0 192.0 256.0 320.0 384.0 448.0 512.0 576.0 640.0 704.0 768.0 832.0 896.0 960.0 1024.0
ITERATION (N)


DIGITAL FILTER FREQUENCY RESPONSE
SIMULATION INPUT/OUTPUT VS ITERATION
FREQUENCY - 575 HZ
MAGNITUDE
2.0
1.5
1.0
0.5
0.0
-0.5
-1.0
-1.5
-2.0
0.0 64.0 128.0 192.0 256.0 320.0 384.0 448.0 512.0 576.0 640.0 704.0 768.0 832.0 896.0 960.0 1024.0
ITERATION (N)


DIGITAL FILTER FREQUENCY RESPONSE
SIMULATION INPUT/OUTPUT VS ITERATION
FREQUENCY - 590.825 HZ
MAGNITUDE
2.0
1.5
1.0
0.5
0.0
-0.5
-1.0
-1.5
-2.0
0.0 64.0 128.0 192.0 256.0 320.0 384.0 448.0 512.0 576.0 640.0 704.0 768.0 832.0 896.0 960.0 1024.0
ITERATION (N)


DIGITAL FILTER FREQUENCY RESPONSE
SIMULATION INPUT/OUTPUT VS ITERATION
FREQUENCY - 600 HZ
MAGNITUDE
2.0
1.5
1.0
0.5
0.0
-0.5
-1.0
-1.5
-2.0
0.0 64.0 128.0 192.0 256.0 320.0 384.0 448.0 512.0 576.0 640.0 704.0 768.0 832.0 896.0 960.0 1024.0
ITERATION (N)


DIGITAL FILTER FREQUENCY RESPONSE
SIMULATION INPUT/OUTPUT VS ITERATION
FREQUENCY - 625 HZ
MAGNITUDE
2.0
1.5
1.0
0.5
0.0
-0.5
-1.0
-1.5
-2.0
0.0 64.0 128.0 192.0 256.0 320.0 384.0 448.0 512.0 576.0 640.0 704.0 768.0 832.0 896.0 960.0 1024.0
ITERATION (N)


DIGITAL FILTER FREQUENCY RESPONSE
SIMULATION INPUT/OUTPUT VS ITERATION
FREQUENCY - 700 HZ
MAGNITUDE
2.0
1.5
1.0
0.5
0.0
-0.5
-1.0
-1.5
-2.0
0.0 64.0 128.0 192.0 256.0 320.0 384.0 448.0 512.0 576.0 640.0 704.0 768.0 832.0 896.0 960.0 1024.0
ITERATION (N)

ASSEMBLER INVOKED BY: AS2920 SM3X4

```
 LINE  LOC OBJECT SOURCE STATEMENT

    1              ;
    2              ;            APPENDIX H
    3              ;
    4              ; 2920 ASSEMBLY LANGUAGE PROGRAM SM3X2
    5              ; 600 HZ CENTER FREQUENCY BAND-PASS FILTER
    6              ; FOURTH ORDER ELLIPTIC FILTER
    7              ; AUTHOR: LT D. W. JORDAN
    8              ;
    9              ; CLEAR DAR REGISTER
   10    0 40C6EF     LDA DAR,KP0
   11              ;
   12              ; INPUT ANALOG SAMPLE TO SAMPLE/HOLD
   13              ; INPUT MUST BE LESS THAN 1.0 VOLTS
   14    1 0000EF     IN0
   15    2 0000EF     IN0
   16    3 0000EF     IN0
   17    4 0000EF     IN0
   18    5 4000EF     NOP
   19    6 4000EF     NOP
   20    7 4000EF     NOP
   21              ;
   22              ; BEGIN ANALOG TO DIGITAL CONVERSION
   23              ; DIGITAL SAMPLE WILL RESIDE IN DAR REGISTER
   24    8 6000EF     OUT3
   25    9 4000EF     NOP
   26   10 4000EF     NOP
   27   11 4000EF     NOP
   28   12 7100EF     OUT7
   29   13 4000EF     NOP
   30   14 4000EF     NOP
   31   15 4000EF     NOP
   32   16 6100EF     OUT6
   33   17 4000EF     NOP
   34   18 4000EF     NOP
   35   19 4000EF     NOP
   36   20 5100EF     OUT5
   37   21 4000EF     NOP
   38   22 4000EF     NOP
   39   23 4000EF     NOP
   40   24 4100EF     OUT4
   41   25 4000EF     NOP
   42   26 4000EF     NOP
   43   27 4000EF     NOP
   44   28 3100EF     OUT3
   45   29 4000EF     NOP
   46   30 4000EF     NOP
   47   31 4000EF     NOP
   48   32 2100EF     OUT2
   49   33 4000EF     NOP
   50   34 4000EF     NOP
   51   35 4000EF     NOP
   52   36 1100EF     OUT1
```

```
 LINE  LOC OBJECT SOURCE STATEMENT

   53   37 4000EF     NOP
   54   38 4000EF     NOP
   55   39 4000EF     NOP
   56   40 0100EF     OUT0
   57   41 4000EF     NOP
   58   42 4000EF     NOP
   59   43 4000EF     NOP
   60                 ;
   61                 ; SCALE DOWN DIGITAL INPUT BY A FACTOR OF 64
   62   44 4066AE     LDA DAP,DAP,A06
   63                 ;
   64                 ; LOAD SCALED INPUT FROM DAP INTO X11
   65   45 402EEF     LDA X11,DAP
   66                 ;
   67                 ; INITIALIZE Y11
   68   46 4032FF     LDA Y11,KP0
   69                 ;
   70                 ; PERFORM FIRST STAGE DIFFERENCE EQUATION COMPUTATION
   71                 ; Y11=N13*X11-N12*X12+N11*X13+D12*Y12-D11*Y13
   72                 ;
   73                 ; Y11=N13*X11
   74                 ; (N13=1.0000000)
   75                 ;
   76   47 4400EF     LDA PPD,X11
   77                 ;
   78   48 4200FD     ADD Y11,PPD
   79                 ;
   80                 ; Y11=Y11-N12*X12
   81                 ; (N12=1.749875)
   82                 ;
   83   49 4608EF     LDA PPD,X12,P00
   84   50 46080C     ADD PPD,X12,P01
   85   51 46084C     ADD PPD,X12,P03
   86   52 46086C     ADD PPD,X12,P04
   87   53 46088C     ADD PPD,X12,P05
   88   54 4608AC     ADD PPD,X12,P06
   89   55 4608CC     ADD PPD,X12,P07
   90   56 4608EC     ADD PPD,X12,P08
   91   57 46080D     ADD PPD,X12,P09
   92   58 46082D     ADD PPD,X12,P10
   93   59 46084D     ADD PPD,X12,P11
   94   60 46086D     ADD PPD,X12,P12
   95   61 46088D     ADD PPD,X12,P13
   96                 ;
   97   62 4200FB     SUB Y11,PPD
   98                 ;
   99                 ; Y11=Y11+N11*X13
  100                 ; (N11=1.001585)
  101                 ;
  102   63 4000EF     LDA PPD,X13,P00
  103   64 40002D     ADD PPD,X13,P10
  104   65 40004D     ADD PPD,X13,P11
  105   66 40008D     ADD PPD,X13,P13
  106                 ;
```

```
 LINE  LOC OBJECT SOURCE STATEMENT

  107   67 4800FD     ADD Y11,PPD
  108                :
  109                ; Y11=Y11+D12*Y12
  110                ; (D12=1.815550)
  111                ;
  112   68 4008EF     LDA PPD,Y12,P00
  113   69 400800     ADD PPD,Y12,P01
  114   70 400820     ADD PPD,Y12,P02
  115   71 400860     ADD PPD,Y12,P04
  116   72 400801     ADD PPD,Y12,P09
  117   73 400821     ADD PPD,Y12,P10
  118   74 400881     ADD PPD,Y12,P13
  119                ;
  120   75 4800FD     ADD Y11,PPD
  121                ;
  122                ; Y11=Y11-D11*Y13
  123                ; (D11=0.990601)
  124                ;
  125   76 4E000E     LDA PPD,Y13,P01
  126   77 4E0020     ADD PPD,Y13,P02
  127   78 4E0040     ADD PPD,Y13,P03
  128   79 4E0060     ADD PPD,Y13,P04
  129   80 4E0080     ADD PPD,Y13,P05
  130   81 4E00A0     ADD PPD,Y13,P06
  131   82 4E00E0     ADD PPD,Y13,P08
  132   83 4E0001     ADD PPD,Y13,P09
  133   84 4E0061     ADD PPD,Y13,P12
  134   85 4E0081     ADD PPD,Y13,P13
  135                ;
  136   86 4200FB     SUB Y11,PPD
  137                ;
  138                ; INITIALIZE Y21
  139   87 4492FF     LDA Y21,KP0
  140                ;
  141                ; PERFORM SECOND STAGE DIFFERENCE EQUATION COMPUTATION
  142                ; Y21=N23*Y11-N22*X22+N21*X23+D22*Y22-D21*Y23
  143                ;
  144                ; Y21=N23*Y11
  145                ; (N23=1.000000)
  146                ;
  147   88 4408EF     LDA PPD,Y11
  148                ;
  149   89 4610FD     ADD Y21,PPD
  150                ;
  151                ; Y21=Y21-N22*X22
  152                ; (N22=1.877905)
  153                ;
  154   90 4420EF     LDA PPD,X22,P00
  155   91 442000     ADD PPD,X23,P01
  156   92 442020     ADD PPD,X22,P02
  157   93 442040     ADD PPD,X22,P03
  158   94 442001     ADD PPD,X22,P09
  159   95 442041     ADD PPD,X22,P11
  160   96 442061     ADD PPD,X22,P12
```

```
 LINE  LOC OBJECT SOURCE STATEMENT

  161   97 44208D     ADD PPD,X22,R13
  162              ;
  163   98 4610FB     SUB Y21,PPD
  164              ;
  165              ; Y21=Y21+N21*Y22
  166              ; (N21=0.9981690)
  167              ;
  168   99 44280E     LDA PPD,X23,R01
  169  100 44282C     ADD PPD,X23,R02
  170  101 44284C     ADD PPD,X23,R03
  171  102 44286C     ADD PPD,X23,R04
  172  103 44288C     ADD PPD,X23,R05
  173  104 4428AC     ADD PPD,X23,R06
  174  105 4428CC     ADD PPD,X23,R07
  175  106 4428EC     ADD PPD,X23,R08
  176  107 44280D     ADD PPD,X23,R09
  177  108 44288D     ADD PPD,X23,R13
  178              ;
  179  109 4610FD     ADD Y21,PPD
  180              ;
  181              ; Y21=Y21+I22*Y22
  182              ; (I22=1.8079446)
  183              ;
  184  110 4620EF     LDA PPD,Y22,R00
  185  111 46200C     ADD PPD,Y22,R01
  186  112 46202C     ADD PPD,Y22,R02
  187  113 46208C     ADD PPD,Y22,R05
  188  114 4620AC     ADD PPD,Y22,R06
  189  115 4620CC     ADD PPD,Y22,R07
  190  116 4620EC     ADD PPD,Y22,R08
  191  117 46204D     ADD PPD,Y22,R11
  192  118 46206D     ADD PPD,Y22,R12
  193  119 46208D     ADD PPD,Y22,R13
  194              ;
  195  120 4610FD     ADD Y21,PPD
  196              ;
  197              ; Y21=Y21-I21*Y23
  198              ; (I21=0.982544)
  199              ;
  200  121 46290E     LDA PPD,Y23,R01
  201  122 46282C     ADD PPD,Y23,R02
  202  123 46284C     ADD PPD,Y23,R03
  203  124 46286C     ADD PPD,Y23,R04
  204  125 46288C     ADD PPD,Y23,R05
  205  126 4628DC     ADD PPD,Y23,R07
  206  127 4628EC     ADD PPD,Y23,R08
  207  128 46280D     ADD PPD,Y23,R09
  208  129 46288D     ADD PPD,Y23,R13
  209              ;
  210  130 4610FB     SUB Y21,PPD
  211              ;
  212              ; LOAD Y21 INTO IAR FOR OUTPUT
  213  131 4A4CEF     LDA IAR,Y21,R00
  214              ;
```

```
8818-II 2980 ASSEMBLER V1.0                                    PAGE    5


  LINE  LOC OBJECT SOURCE STATEMENT

   215              ; MULTIPLY OUTPUT BY A FACTOR OF 4
   216  132 406AAF     LDA DAP,DAP,L02
   217  133 4000EF     NOP
   218  134 4000EF     NOP
   219  135 4000EF     NOP
   220  136 4000EF     NOP
   221  137 4000EF     NOP
   222  138 4000EF     NOP
   223              ;
   224              ; OUTPUT VALUE IN DAP TO CHANNEL 0
   225  139 8000EF     OUT0
   226  140 8000EF     OUT0
   227  141 8000EF     OUT0
   228  142 8000EF     OUT0
   229              ;
   230              ; SERIAL REGISTER SHIFT FOR NEXT PROGRAM PASS
   231  143 4660FF     LDA Y23,Y22,A00
   232  144 4E48EF     LDA Y22,Y21,A00
   233  145 4C18EF     LDA Y13,Y12,A00
   234  146 4018FF     LDA Y12,Y11,A00
   235  147 4060FF     LDA X23,X22,A00
   236  148 4048EF     LDA X22,Y11,A00
   237  149 4218EF     LDA X13,X12,A00
   238  150 4400FF     LDA X12,X11,A00
   239  151 4000EF     NOP
   240  152 4000EF     NOP
   241  153 4000EF     NOP
   242  154 4000EF     NOP
   243  155 4000EF     NOP
   244  156 4000EF     NOP
   245  157 4000EF     NOP
   246  158 4000EF     NOP
   247  159 4000EF     NOP
   248  160 4000EF     NOP
   249  161 4000EF     NOP
   250  162 4000EF     NOP
   251  163 4000EF     NOP
   252  164 4000EF     NOP
   253  165 4000EF     NOP
   254  166 4000EF     NOP
   255  167 4000EF     NOP
   256  168 4000EF     NOP
   257  169 4000EF     NOP
   258  170 4000EF     NOP
   259  171 4000EF     NOP
   260  172 4000EF     NOP
   261  173 4000EF     NOP
   262  174 4000EF     NOP
   263  175 4000EF     NOP
   264  176 4000EF     NOP
   265  177 4000EF     NOP
   266  178 4000EF     NOP
   267  179 4000EF     NOP
   268  180 4000EF     NOP
```

```
ISIS-II 8920 ASSEMBLER V1.0                                    PAGE    F


  LINE  LOC OBJECT SOURCE STATEMENT

   269  181 4000EF      NOP
   270  182 4000EF      NOP
   271  183 4000EF      NOP
   272  184 4000EF      NOP
   273  185 4000EF      NOP
   274  186 4000EF      NOP
   275  187 4000EF      NOP
   276                ;
   277                ; THIS IS THE FINAL FOUR INSTRUCTION SEGMENT
   278  188 5000EF      EOP
   279  189 4000EF      NOP
   280  190 4000EF      NOP
   281  191 4000EF      NOP
   282                  END

SYMBOL:                          VALUE:

X11                                    0
Y11                                    1
PRD                                    2
X12                                    3
X13                                    4
Y12                                    5
Y13                                    6
Y21                                    7
X22                                    8
X23                                    9
Y22                                   10
Y23                                   11

ASSEMBLY COMPLETE
ERRORS    =     0
WARNINGS  =     0
RAMSIZE   =    12
ROMSIZE   =   192
```

FILE: CTRANS2 FORTRAN A1

```
C*************************************************************************CTR00010
C                                                                         *CTR00020
C                              APPENDIX I                                 *CTR00030
C                                                                         *CTR00040
C                        FORTRAN PROGRAM CTRANS2                          *CTR00050
C                                                                         *CTR00060
C                                                                         *CTR00070
C      THIS PROGRAM PERFORMS A TRANSFORMATION OF THE COEFFICIENTS         *CTR00080
C      OF TWO SECOND ORDER POLYNOMIAL STAGES FOR 2920 IMPLEMENTATION      *CTR00090
C                                                                         *CTR00100
C*************************************************************************CTR00110
C VARIABLE DECLARATIONS                                                    CTR00120
      REAL N2(2,3),D2(2,3),N2TX,D2TX,TRIAL,TWOVAL,FKM1                     CTR00130
      REAL N2B(2,3),D2B(2,3),N2P(2,3),D2P(2,3)                             CTR00140
      !NTEGER N2T(2,3,14),D2T(2,3,14)                                      CTR00150
      INTEGER JM1,JX,KM1                                                   CTR00160
C ACTUAL FIRST STAGE COEFFICIENTS TO BE TRANSFORMED                        CTR00170
      N2(1,3)=1.                                                           CTR00180
      N2(1,2)=-1.7499                                                      CTR00190
      N2(1,1)=1.0017                                                       CTR00200
      D2(1,3)=1.                                                           CTR00210
      D2(1,2)=-1.8156                                                      CTR00220
      D2(1,1)=0.99068                                                      CTR00230
C ACTUAL SECOND STAGE COEFFICIENTS TO BE TRANSFORMED                       CTR00240
      N2(2,3)=1.                                                           CTR00250
      N2(2,2)=-1.8779                                                      CTR00260
      N2(2,1)=0.99817                                                      CTR00270
      D2(2,3)=1.                                                           CTR00280
      D2(2,2)=-1.8095                                                      CTR00290
      D2(2,1)=0.98255                                                      CTR00300
C INITIALIZE BINARY COEFFICIENT MATRIX                                     CTR00310
      DO 10 I=1,2                                                          CTR00320
         DO 12 J=1,3                                                       CTR00330
            N2B(I,J)=0.                                                    CTR00340
            D2B(I,J)=0.                                                    CTR00350
            N2P(I,J)=0.                                                    CTR00360
            D2P(I,J)=0.                                                    CTR00370
            DO 14 K=1,13                                                   CTR00380
               N2T(I,J,K)=0                                                CTR00390
               D2T(I,J,K)=0                                                CTR00400
   14       CONTINUE                                                       CTR00410
   12    CONTINUE                                                          CTR00420
   10 CONTINUE                                                           CTR00430
C PERFORM COEFFICIENT TRANSFORMATION TO BINARY 2920 REPRESENTATION         CTR00440
C NUMERATOR TERMS                                                          CTR00450
      DO 20 I=1,2                                                          CTR00460
         DO 22 J=1,3                                                       CTR00470
            N2TX=ABS(N2(I,J))                                              CTR00480
            IF (N2TX-1.0) 221,222,223                                      CTR00490
  222          N2T(I,J,1)=1                                                CTR00500
               N2B(I,J)=1.0                                                CTR00510
               GO TO 22                                                    CTR00520
  223          N2T(I,J,1)=1                                                CTR00530
               N2B(I,J)=1.0                                                CTR00540
               N2TX=N2TX-1.                                                CTR00550
```

FILE: CTRANS2 FORTRAN A1

```
  221          DO 24 K=2,14                                                  CTR00560
                  FKM1=FLOAT(K-1)                                            CTR00570
                  TWOVAL=1./(2.**FKM1)                                       CTR00580
                  TRIAL=N2TX-TWOVAL                                          CTR00590
                  IF (TRIAL) 25,242,243                                      CTR00600
  242                N2T(I,J,K)=1                                          CTR00610
                     N2B(I,J)=N2B(I,J)+TWOVAL                              CTR00620
                     GO TO 20                                              CTR00630
  243                N2T(I,J,K)=1                                          CTR00640
                     N2B(I,J)=N2B(I,J)+TWOVAL                              CTR00650
                     N2TX=TRIAL                                            CTR00660
                     GO TO 24                                              CTR00670
   25                IF (K.EQ.14) N2T(I,J,K)=0                             CTR00680
   24          CONTINUE                                                    CTR00690
   22    CONTINUE                                                          CTR00700
   20 CONTINUE                                                             CTR00710
C DENOMINATOR TERMS                                                        CTR00720
      DO 30 I=1,2                                                          CTR00730
         DO 32 J=1,3                                                       CTR00740
            D2TX=ABS(D2(I,J))                                              CTR00750
            IF (D2TX-1.0) 321,322,323                                      CTR00760
  322          D2T(I,J,1)=1                                                CTR00770
               D2B(I,J)=1.0                                                CTR00780
               GO TO 32                                                    CTR00790
  323          D2T(I,J,1)=1                                                CTR00800
               D2B(I,J)=1.0                                                CTR00810
               D2TX=D2TX-1.                                                CTR00820
  321          DO 34 K=2,14                                                CTR00830
                  FKM1=FLOAT(K-1)                                          CTR00840
                  TWOVAL=1./(2.**FKM1)                                     CTR00850
                  TRIAL=D2TX-TWOVAL                                        CTR00860
                  IF (TRIAL) 35,342,343                                    CTR00870
  342                D2T(I,J,K)=1                                          CTR00880
                     D2B(I,J)=D2B(I,J)+TWOVAL                              CTR00890
                     GO TO 30                                              CTR00900
  343                D2T(I,J,K)=1                                          CTR00910
                     D2B(I,J)=D2B(I,J)+TWOVAL                              CTR00920
                     D2TX=TRIAL                                            CTR00930
                     GO TO 34                                              CTR00940
   35                IF (K.EQ.14) D2T(I,J,K)=0                             CTR00950
   34          CONTINUE                                                    CTR00960
   32    CONTINUE                                                          CTR00970
   30 CONTINUE                                                             CTR00980
C PRINT OUTPUT HEADING                                                     CTR00990
      WRITE(4,80)                                                          CTR01000
   80 FORMAT(' PROGRAM CTRANS2 OUTPUT',//,                                 CTR01010
     &' FOURTH ORDER DIGITAL FILTER 2920 BINARY EQUIVALENTS',/,            CTR01020
     &'   (TWO CASCADED SECOND ORDER SECTIONS)',///)                       CTR01030
C PRINT FIRST STAGE TRANSFER FUNCTION                                      CTR01040
      WRITE(4,81)N2(1,3),N2(1,2),N2(1,1),D2(1,3),D2(1,2),D2(1,1)           CTR01050
   81 FORMAT(' FIRST STAGE TRANSFER FUNCTION',//,                          CTR01060
     &' ',E14.6,' + ',E14.6,' Z**-1 + ',E14.6,' Z**-2',/,                  CTR01070
     &'   ----------------------------------------------------',/,CTR01080
     &' ',E14.6,' + ',E14.6,' Z**-1 + ',E14.6,' Z**-2')                    CTR01090
      WRITE(4,810)                                                         CTR01100
```

```
  810 FORMAT(///,' BINARY REPRESENTATION OF NUMERATOR COEFFICIENTS',//, CTR01110
     &'        R00 R01 R02 R03 R04 R05 R06 R07 R08 R09 R10 R11 R12 R13' CTR01120
     &,/)                                                              CTR01130
      DO 811 J=1,3                                                     CTR01140
         JX=4-J                                                        CTR01150
         N2P(1,JX)=N2B(1,JX)/ABS(N2(1,JX))                             CTR01160
         WRITE(4,812)JX,N2(1,JX)                                       CTR01170
  812    FORMAT(' N2(1,',I1,') = ',F9.6)                               CTR01180
         WRITE(4,814)N2T(1,JX,1),N2T(1,JX,2),N2T(1,JX,3),N2T(1,JX,4),  CTR01190
     &N2T(1,JX,5),N2T(1,JX,6),N2T(1,JX,7),N2T(1,JX,8),N2T(1,JX,9),     CTR01200
     &N2T(1,JX,10),N2T(1,JX,11),N2T(1,JX,12),N2T(1,JX,13),N2T(1,JX,14), CTR01210
     &N2B(1,JX),N2P(1,JX)                                              CTR01220
  814    FORMAT(9X,14(I1,3X),/,' ABSOLUTE BINARY EQUIVALENT = '        CTR01230
     &,F9.6,/,' (THIS IS ',F9.6,' OF THE ACTUAL VALUE)',/)             CTR01240
  811 CONTINUE                                                         CTR01250
      WRITE(4,815)                                                     CTR01260
  815 FORMAT(//,' BINARY REPRESENTATION OF DENOMINATOR COEFFICIENTS',//,CTR01270
     &'        R00 R01 R02 R03 R04 R05 R06 R07 R08 R09 R10 R11 R12 R13' CTR01280
     &,/)                                                              CTR01290
      DO 816 J=1,3                                                     CTR01300
         JX=4-J                                                        CTR01310
         D2P(1,JX)=D2B(1,JX)/ABS(D2(1,JX))                             CTR01320
         WRITE(4,817)JX,D2(1,JX)                                       CTR01330
  817    FORMAT(' D2(1,',I1,') = ',F9.6)                               CTR01340
         WRITE(4,819)D2T(1,JX,1),D2T(1,JX,2),D2T(1,JX,3),D2T(1,JX,4),  CTR01350
     &D2T(1,JX,5),D2T(1,JX,6),D2T(1,JX,7),D2T(1,JX,8),D2T(1,JX,9),     CTR01360
     &D2T(1,JX,10),D2T(1,JX,11),D2T(1,JX,12),D2T(1,JX,13),D2T(1,JX,14), CTR01370
     &D2B(1,JX),D2P(1,JX)                                              CTR01380
  819    FORMAT(9X,14(I1,3X),/,' ABSOLUTE BINARY EQUIVALENT = '        CTR01390
     &,F9.6,/,' (THIS IS ',F9.6,' OF THE ACTUAL VALUE)',//)            CTR01400
  816 CONTINUE                                                         CTR01410
C PRINT SECOND STAGE TRANSFER FUNCTION                                 CTR01420
      WRITE(4,82)N2(2,3),N2(2,2),N2(2,1),D2(2,3),D2(2,2),D2(2,1)       CTR01430
   82 FORMAT('1',' SECOND STAGE TRANSFER FUNCTION',//,                 CTR01440
     &E14.6,' + ',E14.6,' Z**-1 + ',E14.6,' Z**-2',/,                  CTR01450
     &'  ----------------------------------------------------',/,      CTR01460
     &E14.6,' + ',E14.6,' Z**-1 + ',E14.6,' Z**-2')                    CTR01470
      WRITE(4,820)                                                     CTR01480
  820 FORMAT(//,' BINARY REPRESENTATION OF NUMERATOR COEFFICIENTS',//,  CTR01490
     &'        R00 R01 R02 R03 R04 R05 R06 R07 R08 R09 R10 R11 R12 R13' CTR01500
     &,/)                                                              CTR01510
      DO 821 J=1,3                                                     CTR01520
         JX=4-J                                                        CTR01530
         N2P(2,JX)=N2B(2,JX)/ABS(N2(2,JX))                             CTR01540
         WRITE(4,822)JX,N2(1,JX)                                       CTR01550
  822    FORMAT(' N2(2,',I1,') = ',F9.6)                               CTR01560
         WRITE(4,824)N2T(2,JX,1),N2T(2,JX,2),N2T(2,JX,3),N2T(2,JX,4),  CTR01570
     &N2T(2,JX,5),N2T(2,JX,6),N2T(2,JX,7),N2T(2,JX,8),N2T(2,JX,9),     CTR01580
     &N2T(2,JX,10),N2T(2,JX,11),N2T(2,JX,12),N2T(2,JX,13),N2T(2,JX,14), CTR01590
     &N2B(2,JX),N2P(2,JX)                                              CTR01600
  824    FORMAT(9X,14(I1,3X),/,' ABSOLUTE BINARY EQUIVALENT = '        CTR01610
     &,F9.6,/,' (THIS IS ',F9.6,' OF THE ACTUAL VALUE)',//)            CTR01620
  821 CONTINUE                                                         CTR01630
      WRITE(4,825)                                                     CTR01640
  825 FORMAT(/,' BINARY REPRESENTATION OF DENOMINATOR COEFFICIENTS',//, CTR01650
```

FILE: CTRANS2 FORTRAN A1

```
     &'         R00 R01 R02 R03 R04 R05 R06 R07 R08 R09 R10 R11 R12 R13' CTR01660
     &,/)                                                                 CTR01670
      DO 826 J=1,3                                                        CTR01680
         JX=4-J                                                           CTR01690
         D2P(2,JX)=D2B(2,JX)/ABS(D2(2,JX))                                CTR01700
         WRITE(4,827)JX,D2(2,JX)                                          CTR01710
 827     FORMAT(' D2(2,',I1,') = ',F9.6)                                  CTR01720
         WRITE(4,829)D2T(2,JX,1),D2T(2,JX,2),D2T(2,JX,3),D2T(2,JX,4),     CTR01730
    &D2T(2,JX,5),D2T(2,JX,6),D2T(2,JX,7),D2T(2,JX,8),D2T(2,JX,9),         CTR01740
    &D2T(2,JX,10),D2T(2,JX,11),D2T(2,JX,12),D2T(2,JX,13),D2T(2,JX,14),    CTR01750
    &D2B(2,JX),D2P(2,JX)                                                 CTR01760
 829     FORMAT(9X,14(I1,3X),/,' ABSOLUTE BINARY EQUIVALENT = '           CTR01770
    &,F9.6,/,' (THIS IS ',F9.6,' OF THE ACTUAL VALUE)',//)               CTR01780
 826 CONTINUE                                                            CTR01790
     STOP                                                                CTR01800
     END                                                                 CTR01810
```

```
PROGRAM CTRANS2 OUTPUT

FOURTH ORDER DIGITAL FILTER 2920 BINARY EQUIVALENTS
  (TWO CASCADED SECOND ORDER SECTIONS)


FIRST STAGE TRANSFER FUNCTION

  0.100000E+01 +  -0.174990E+01 Z**-1 +   0.100170E+01 Z**-2
  ---------------------------------------------------------
  0.100000E+01 +  -0.181560E+01 Z**-1 +   0.990680E+00 Z**-2


BINARY REPRESENTATION OF NUMERATOR COEFFICIENTS

       R00 R01 R02 R03 R04 R05 R06 R07 R08 R09 R10 R11 R12 R13

N2(1,3) =  1.000000
         1   0   0   0   0   0   0   0   0   0   0   0   0   0
ABSOLUTE BINARY EQUIVALENT =  1.000000
(THIS IS  1.000000 OF THE ACTUAL VALUE)

N2(1,2) = -1.749900
         1   1   0   1   1   1   1   1   1   1   1   1   1   1
ABSOLUTE BINARY EQUIVALENT =  1.749875
(THIS IS  0.999986 OF THE ACTUAL VALUE)

N2(1,1) =  1.001700
         1   0   0   0   0   0   0   0   0   0   1   1   0   1
ABSOLUTE BINARY EQUIVALENT =  1.001585
(THIS IS  0.999885 OF THE ACTUAL VALUE)


BINARY REPRESENTATION OF DENOMINATOR COEFFICIENTS

       R00 R01 R02 R03 R04 R05 R06 R07 R08 R09 R10 R11 R12 R13

D2(1,3) =  1.000000
         1   0   0   0   0   0   0   0   0   0   0   0   0   0
ABSOLUTE BINARY EQUIVALENT =  1.000000
(THIS IS  1.000000 OF THE ACTUAL VALUE)


D2(1,2) = -1.815600
         1   1   1   0   1   0   0   0   0   1   1   0   0   1
ABSOLUTE BINARY EQUIVALENT =  1.815550
(THIS IS  0.999972 OF THE ACTUAL VALUE)


D2(1,1) =  0.990680
         0   1   1   1   1   1   1   0   1   1   0   0   1   1
ABSOLUTE BINARY EQUIVALENT =  0.990601
(THIS IS  0.999920 OF THE ACTUAL VALUE)
```

SECOND STAGE TRANSFER FUNCTION

$$\frac{0.100000E+01 + -0.187790E+01\ Z^{**-1} + 0.998170E+00\ Z^{**-2}}{0.100000E+01 + -0.180950E+01\ Z^{**-1} + 0.982550E+00\ Z^{**-2}}$$

BINARY REPRESENTATION OF NUMERATOR COEFFICIENTS

	R00	R01	R02	R03	R04	R05	R06	R07	R08	R09	R10	R11	R12	R13
N2(2,3) = 1.000000	1	0	0	0	0	0	0	0	0	0	0	0	0	0
ABSOLUTE BINARY EQUIVALENT = 1.000000														
(THIS IS 1.000000 OF THE ACTUAL VALUE)														
N2(2,2) = -1.749900	1	1	1	1	0	0	0	0	0	1	0	1	1	1
ABSOLUTE BINARY EQUIVALENT = 1.877805														
(THIS IS 0.999949 OF THE ACTUAL VALUE)														
N2(2,1) = 1.001700	0	1	1	1	1	1	1	1	1	1	0	0	0	1
ABSOLUTE BINARY EQUIVALENT = 0.998169														
(THIS IS 0.999999 OF THE ACTUAL VALUE)														

BINARY REPRESENTATION OF DENOMINATOR COEFFICIENTS

	R00	R01	R02	R03	R04	R05	R06	R07	R08	R09	R10	R11	R12	R13
D2(2,3) = 1.000000	1	0	0	0	0	0	0	0	0	0	0	0	0	0
ABSOLUTE BINARY EQUIVALENT = 1.000000														
(THIS IS 1.000000 OF THE ACTUAL VALUE)														
D2(2,2) = -1.809500	1	1	1	0	0	1	1	1	1	0	0	1	1	1
ABSOLUTE BINARY EQUIVALENT = 1.809446														
(THIS IS 0.999970 OF THE ACTUAL VALUE)														
D2(2,1) = 0.982550	0	1	1	1	1	1	0	1	1	1	0	0	0	1
ABSOLUTE BINARY EQUIVALENT = 0.982544														
(THIS IS 0.999994 OF THE ACTUAL VALUE)														

FILE: FIR4P FORTRAN A1

```
C*************************************************************************FIR00010
C                                                                         *FIR00020
C                              APPENDIX J                                 *FIR00030
C                                                                         *FIR00040
C                         FORTRAN PROGRAM FIR4                            *FIR00050
C                      ADAPTIVE TRANSVERSAL FILTER                        *FIR00060
C                                                                         *FIR00070
C  THIS PROGRAM WILL OBTAIN THE OPTIMAL FILTER WEIGHTS FOR THE FIR        *FIR00080
C  FILTER OF ORDER FOUR.  THE ALGORITHM BEGINS WITH A TRIAL REGION        *FIR00090
C  (-40,40) FOR EACH OF THE WEIGHTS AND DOES A SUCCESSIVE ITERATION       *FIR00100
C  OF THE ERROR FUNCTION WHILE TRANSFORMING THE WEIGHTS TO OPTIMAL        *FIR00110
C  VALUES.  WHEN THE WEIGHTS CONVERGE TO OPTIMAL VALUES THEN THE          *FIR00120
C  ITERATION STOPS.  THEN WE MAY TEST THE ADEQUACY OF THE WEIGHTS         *FIR00130
C  SO OBTAINED THROUGH SUBSEQUENT SIMULATION IN THE FORTRAN PROGRAM       *FIR00140
C  FIR4SIM WHICH FOLLOWS AS APPPENDIX K.                                  *FIR00150
C                                                                         *FIR00160
C*************************************************************************FIR00170
      REAL WS(4),WU(4),WL(4),R,JE                                          FIR00180
      INTEGER NTA,NPR,NAV,NV,IP                                            FIR00190
C  WS(I) IS THE STARTING GUESS                                            FIR00200
      WS(1)=.1                                                             FIR00210
      WS(2)=1.                                                             FIR00220
      WS(3)=1.                                                             FIR00230
      WS(4)=1.                                                             FIR00240
C  WL(I) IS THE LOWER LIMIT FOR THE I'TH VARIABLE                         FIR00250
C  WU(I) IS THE UPPER LIMIT FOR THE I'TH VARIABLE                         FIR00260
      WL(1)=-14.                                                           FIR00270
      WU(1)=14.                                                            FIR00280
      WL(2)=-14.                                                           FIR00290
      WU(2)=14.                                                            FIR00300
      WL(3)=-14.                                                           FIR00310
      WU(3)=14.                                                            FIR00320
      WL(4)=-14.                                                           FIR00330
      WU(4)=14.                                                            FIR00340
C  A DESCRIPTION OF THE FOLLOWING PARAMETERS IS DISCUSSED IN BOXPLX       FIR00350
      R=9./13.                                                             FIR00360
      NTA=1400                                                             FIR00370
      NPR=100                                                              FIR00380
      NAV=0                                                                FIR00390
      NV=4                                                                 FIR00400
      IP=0                                                                 FIR00410
C  PERFORM ITERATION ROUTINE FOR WEIGHT OPTIMIZATION                      FIR00420
      CALL BOXPLX(NV,NAV,NPR,NTA,R,WS,IP,WU,WL,YMN,IER)                    FIR00430
      WRITE (6,25)                                                         FIR00440
25    FORMAT(1X,' OPTIMAL GAINS',/)                                        FIR00450
      DO 30 I=1,4                                                          FIR00460
30       WRITE(6,40)I,WS(I)                                                FIR00470
40    FORMAT(1X,'W(',I2,')=',F14.7)                                        FIR00480
      STOP                                                                 FIR00490
      END                                                                  FIR00500
C*************************************************************************FIR00510
      SUBROUTINE FIR(XX)                                                   FIR00520
C  SUBROUTINE FIR(XX) SIMULATES THE FIR FILTER                            FIR00530
      COMMON J                                                             FIR00540
      REAL*8 J,W0,W1,W2,W3,X1,X2,X3,INPUT,OUTPUT                           FIR00550
```

FILE: FIR4P FORTRAN A1

```
      DIMENSION XX(4),DESIRE(105)                                        FIR00560
C  INITIAL CONDITIONS                                                    FIR00570
      ETIME=100.                                                         FIR00580
      T=0.0                                                              FIR00590
      ICOUNT=2                                                           FIR00600
C  INITIALIZE THE COST (CUMULATIVE ERROR) FUNCTION                       FIR00610
      J=0.0                                                              FIR00620
C  GAIN COEFFICIENTS TO BE OPTIMIZED                                     FIR00630
      W0=XX(1)                                                           FIR00640
      W1=XX(2)                                                           FIR00650
      W2=XX(3)                                                           FIR00660
      W3=XX(4)                                                           FIR00670
C  SHIFT REGISTERS                                                       FIR00680
      X1=0.0                                                             FIR00690
      X2=0.0                                                             FIR00700
      X3=0.0                                                             FIR00710
C  SIMULATE DESIRED OUTPUT SIGNAL                                        FIR00720
      DO 15 I=1,105                                                      FIR00730
15       DESIRE(I)=-1.0                                                  FIR00740
      DO 16 I=1,11                                                       FIR00750
16       DESIRE(I+44)=1.0                                                FIR00760
C                                                                        FIR00770
      K=1                                                                FIR00780
C  OUTPUT HEADING                                                        FIR00790
C     WRITE(6,99)                                                        FIR00800
C  99 FORMAT(' ','FIR TRANSVERSAL FILTER SIMULATION RESULTS',///,        FIR00810
C    &'  TIME       INPUT       SIMULATED OUTPUT       DESIRED OUTPUT',/) FIR00820
C  LOOP FOR 100 SAMPLE ITERATIONS                                        FIR00830
  200 CONTINUE                                                           FIR00840
C  SIMULATED INPUT SIGNAL                                                FIR00850
      INPUT=SIN(.1*T)*COS(.1*T)*(2.+COS(.1*T))                           FIR00860
C     INPUT=-.0004*T**2+.04*T                                            FIR00870
C  SIMULATED OUTPUT SIGNAL FROM FIR FILTER                               FIR00880
      OUTPUT=W0*INPUT+W1*X1+W2*X2+W3*X3                                  FIR00890
C  WHEN TO PRINTOUT                                                      FIR00900
      IF (ICOUNT.EQ. 2) GO TO 50                                         FIR00910
      GO TO 300                                                          FIR00920
C  PRINTOUT                                                              FIR00930
   50 CONTINUE                                                           FIR00940
C  EASYPLOT OUTPUT OPTION                                                FIR00950
C     WRITE (6,100) T,INPUT,OUTPUT,DESIRE(K)                             FIR00960
C 100 FORMAT(2X,F8.4,1X,F8.4,3X,F8.4,13X,F8.4)                           FIR00970
C  SCREEN OUTPUT OPTION                                                  FIR00980
C     WRITE (6,100) T,INPUT,OUTPUT,DESIRE(K)                             FIR00990
C 100 FORMAT(1X,'TIME=',F7.3,5X,'INPUT=',F8.4,5X,'OUTPUT=',F8.4,5X,      FIR01000
C    &'DESIRED OUTPUT=',F8.4)                                            FIR01010
      ICOUNT=1                                                           FIR01020
C  TEST IF WANT TO STOP                                                  FIR01030
 300  IF (T.GE.ETIME) GO TO 400                                          FIR01040
C  JE=ERROR FUNCTION                                                     FIR01050
      JE=(OUTPUT-DESIRE(K))**2                                           FIR01060
C  J=COST FUNCTION (CUMULATIVE ERROR)                                    FIR01070
      J=J+JE                                                             FIR01080
C  STEP SIZE DELT                                                        FIR01090
      DELT=1.0                                                           FIR01100
```

```
      T=T+DELT                                                          FIR01110
      K=K+1                                                             FIR01120
      ICOUNT=ICOUNT+1                                                   FIR01130
      X3=X2                                                             FIR01140
      X2=X1                                                             FIR01150
      X1=INPUT                                                          FIR01160
      GO TO 200                                                         FIR01170
C  OUTPUT OPTIMAL WEIGHTS                                               FIR01180
 400  WRITE(6,500) JE,W0,W1,W2,W3                                       FIR01190
 500  FORMAT(' ',1X,'  J =',E15.9,2X,                                   FIR01200
     1 'W0=',F15.7,2X,'W1=',F15.7,2X,'W2=',F15.7,2X,'W3=',F15.7)       FIR01210
      RETURN                                                            FIR01220
      END                                                               FIR01230
C     ..................................................................FIR01240
C                                                                       FIR01250
C     SUBROUTINE BOXPLX              (CATEGORY H0)                      FIR01260
C                                                                       FIR01270
C .   PURPOSE                                                           FIR01280
C                                                                       FIR01290
C        BOXPLX IS A SUBROUTINE USED TO SOLVE THE PROBLEM OF LOCATING   FIR01300
C        A MINIMUM (OR MAXIMUM) OF AN ARBITRARY OBJECTIVE FUNCTION      FIR01310
C        SUBJECT TO ARBITRARY EXPLICIT AND/OR IMPLICIT CONSTRAINTS BY   FIR01320
C        THE COMPLEX METHOD OF M.J. BOX.  EXPLICIT CONSTRAINTS ARE      FIR01330
C        DEFINED AS UPPER AND LOWER BOUNDS ON THE INDEPENDENT VARIABLES.FIR01340
C        IMPLICIT CONSTRAINTS MAY BE ARBITRARY FUNCTIONS OF THE VAR-    FIR01350
C        IABLES.  TWO FUNCTION SUBPROGRAMS TO EVALUATE THE OBJECTIVE    FIR01360
C        FUNCTION AND IMPLICIT CONSTRAINTS, RESPECTIVELY, MUST BE       FIR01370
C        SUPPLIED BY THE USER (SEE EXAMPLE BELOW).  BOXPLX ALSO HAS     FIR01380
C        THE OPTION TO PERFORM INTEGER PROGRAMMING, WHERE THE VALUES    FIR01390
C        OF THE INDEPENDENT VARIABLES ARE RESTRICTED TO INTEGERS.       FIR01400
C                                                                       FIR01410
C     USAGE                                                             FIR01420
C                                                                       FIR01430
C        CALL BOXPLX (NV,NAV,NPR,NTA,R,XS,IP,XU,XL,YMN,IER)             FIR01440
C                                                                       FIR01450
C                                                                       FIR01460
C     DESCRIPTION OF PARAMETERS                                         FIR01470
C                                                                       FIR01480
C        NV   AN INTEGER INPUT DEFINING THE NUMBER OF INDEPENDENT       FIR01490
C             VARIABLES OF THE OBJECTIVE FUNCTION TO BE MINIMIZED.      FIR01500
C             NOTE:  MAXIMUM NV + NAV IS PRESENTLY 50.  MAXIMIM NV IS   FIR01510
C             25.  IF THESE LIMITS MUST BE EXCEEDED, PUNCH A SOURCE     FIR01520
C             DECK IN THE USUAL MANNER, AND CHANGE THE DIMENSION        FIR01530
C             STATEMENTS.                                               FIR01540
C                                                                       FIR01550
C        NAV  AN INTEGER INPUT DEFINING THE NUMBER OF AUXILIARY VAR-    FIR01560
C             IABLES THE USER WISHES TO DEFINE FOR HIS OWN CONVENIENCE. FIR01570
C             TYPICALLY HE MAY WISH TO DEFINE THE VALUE OF EACH IMPLICITFIR01580
C             CONSTRAINT FUNCTION AS AN AUXILIARY VARIABLE.  IF THIS    FIR01590
C             IS DONE, THE OPTIONAL OUTPUT FEATURE OF BOXPLX CAN BE     FIR01600
C             USED TO OBSERVE THE VALUES OF THOSE CONSTRAINTS AS THE    FIR01610
C             SOLUTION PROGRESSES.  AUXILIARY VARIABLES, IF USED,       FIR01620
C             SHOULD BE EVALUATED IN FUNCTION KE (DEFINED BELOW).       FIR01630
C             NAV MAY BE ZERO.                                          FIR01640
C                                                                       FIR01650
```

```
C       NPR  INPUT INTEGER CONTROLLING THE FREQUENCY OF OUTPUT DESIRED FIR01660
C            FOR DIAGNOSTIC PURPOSES.  IF NPR .LE. 0, NO OUTPUT WILL BEFIR01670
C            PRODUCED BY BOXPLX.  OTHERWISE, THE CURRENT COMPLEX OF    FIR01680
C            K= 2*NV VERTICES AND THEIR CENTROID WILL BE OUTPUT AFTER  FIR01690
C            EACH NPR PERMISSIBLE TRIALS.  THE NUMBER OF TOTAL TRIALS, FIR01700
C            NUMBER OF FEASIBLE TRIALS, NUMBER OF FUNCTION EVALUATIONS FIR01710
C            AND NUMBER OF IMPLICIT CONSTRAINT EVALUATIONS ARE IN-     FIR01720
C            CLUDED IN THE OUTPUT.                                     FIR01730
C            ADDITIONALLY, (WHEN NPR .GT. 0) THE SAME INFORMATION      FIR01740
C            WILL BE OUTPUT:                                           FIR01750
C                                                                      FIR01760
C              1) IF THE INITIAL POINT IS NOT FEASIBLE,                FIR01770
C              2) AFTER THE FIRST COMPLETE COMPLEX IS GENERATED,        FIR01780
C              3) IF A FEASIBLE VERTEX CANNOT BE FOUND AT SOME TRIAL,  FIR01790
C              4) IF THE OBJECTIVE VALUE OF A VERTEX CANNOT BE MADE    FIR01800
C                 NO-LONGER-WORST.                                     FIR01810
C              5) IF THE LIMIT ON TRIALS (NTA) IS REACHED AND,         FIR01820
C              6) WHEN THE OBJECTIVE FUNCTION HAS BEEN UNCHANGED FOR   FIR01830
C                 2*NV TRIALS, INDICATING A LOCAL MINIMUM HAS BEEN     FIR01840
C                 FOUND.                                               FIR01850
C                                                                      FIR01860
C            IF THE USER WISHES TO TRACE THE PROGRESS OF A SOLUTION,   FIR01870
C            A CHOICE OF NPR = 25, 50 OR 100 IS RECOMMENDED.           FIR01880
C                                                                      FIR01890
C       NTA  INTEGER INPUT OF LIMIT ON THE NUMBER OF TRIALS ALLOWED    FIR01900
C            IN THE CALCULATION.  IF THE USER INPUTS NTA .LE. 0, A     FIR01910
C            DEFAULT VALUE OF 2000 IS USED.  WHEN THIS LIMIT IS REACHEDFIR01920
C            CONTROL RETURNS TO THE CALLING PROGRAM WITH THE BEST      FIR01930
C            ATTAINED OBJECTIVE FUNCTION VALUE IN YMN, AND THE BEST    FIR01940
C            ATTAINED SOLUTION POINT IN XS.                            FIR01950
C                                                                      FIR01960
C       R    A REAL NUMBER INPUT TO DEFINE THE FIRST RANDOM NUMBER     FIR01970
C            USED IN DEVELOPING THE INITIAL COMPLEX OF 2*NV VERTICIES. FIR01980
C            (0. .GT. R .LT. 1.) IF R IS NOT WITHIN THESE BOUNDS,      FIR01990
C            IT WILL BE REPLACED BY 1./3. .                            FIR02000
C                                                                      FIR02010
C       XS   INPUT REAL ARRAY DIMENSIONED AT LEAST NV+NAV. THE FIRST   FIR02020
C            NV MUST CONTAIN A FEASIBLE ORIGIN FOR STARTING THE CAL-   FIR02030
C            CULATION.  THE LAST NAV NEED NOT BE INITIALIZED.  UPON    FIR02040
C            RETURN FROM BOXPLX, THE FIRST NV ELEMENTS OF THE ARRAY    FIR02050
C            CONTAIN THE COORDINATES OF THE MINIMUM OBJECTIVE FUNCTION,FIR02060
C            AND THE REMAINING NAV (NAV .GE. 0) CONTAIN THE VALUES OF  FIR02070
C            THE CORRESPONDING AUXILIARY VARIABLES.                    FIR02080
C                                                                      FIR02090
C       IP   INTEGER INPUT FOR OPTIONAL INTEGER PROGRAMMING.  IF IP=1, FIR02100
C            THE VALUES OF THE INDEPENDENT VARIABLES WILL BE REPLACED  FIR02110
C            WITH INTEGER VALUES (STILL STORED AS REAL*4).             FIR02120
C                                                                      FIR02130
C       XU   A REAL ARRAY DIMENSIONED AT LEAST NV INPUTTING THE UPPER  FIR02140
C            BOUND ON EACH INDEPENDENT VARIABLE, (EACH EXPLICIT CON-   FIR02150
C            STRAINT).  INPUT VALUES ARE SLIGHTLY ALTERED BY BOXPLX.   FIR02160
C                                                                      FIR02170
C       XL   A REAL ARRAY DIMENSIONED AT LEAST NV INPUTTING THE LOWER  FIR02180
C            BOUND ON EACH INDEPENDENT VARIABLE, (EACH EXPLICIT CON-   FIR02190
C            STRAINT).  NOTE:  FOR BOTH XU AND XL CHOOSE REASONABLE    FIR02200
```

```
C             VALUES IF NONE ARE GIVEN, NOT VALUES WHICH ARE MAGNITUDES FIR02210
C             ABOVE OR BELOW THE EXPECTED SOLUTION.  INPUT VALUES ARE   FIR02220
C             SLIGHTLY ALTERED BY BOXPLX.                               FIR02230
C                                                                       FIR02240
C        YMN  THIS OUTPUT IS THE VALUE (REAL*4) OF THE OBJECTIVE FUNC-  FIR02250
C             TION, CORRESPONDING TO THE SOLUTION POINT OUTPUT IN XS.   FIR02260
C                                                                       FIR02270
C        IER  INTEGER ERROR RETURN.  TO BE INTERROGATED UPON RETURN     FIR02280
C             FROM BOXPLX.  IER WILL BE ONE OF THE FOLLOWING:           FIR02290
C                                                                       FIR02300
C               =-1  CANNOT FIND FEASIBLE VERTEX OR FEASIBLE CENTROID   FIR02310
C                    AT THE START OR A RESTART (SEE 'METHOD' BELOW).    FIR02320
C               =0   FUNCTION VALUE UNCHANGED FOR 'N' TRIALS.  (WHERE   FIR02330
C                    N=6*NV+10)  THIS IS THE NORMAL RETURN PARAMETER.   FIR02340
C               =1   CANNOT DEVELOP FEASIBLE VERTEX.                    FIR02350
C               =2   CANNOT DEVELOP A NO-LONGER-WORST VERTEX.           FIR02360
C               =3   LIMIT ON TRIALS REACHED.  (NTA EXCEEDED)           FIR02370
C               NOTE:  VALID RESULTS MAY BE RETURNED IN ANY OF THE      FIR02380
C                    ABOVE CASES.                                       FIR02390
C                                                                       FIR02400
C     EXAMPLE OF USAGE                                                  FIR02410
C                                                                       FIR02420
C        THIS EXAMPLE MINIMIZES THE OBJECTIVE FUNCTION SHOWN IN THE     FIR02430
C        EXTERNAL FUNCTION FE(X).  THERE ARE TWO INDEPENDENT VAR-       FIR02440
C        IABLES X(1) & X(2), AND TWO IMPLICIT CONSTRAINT FUNCTIONS      FIR02450
C        X(3) & X(4) WHICH ARE EVALUATED AS AUXILIARY VARIABLES (SEE    FIR02460
C        EXTERNAL FUNCTION KE(X) ).                                     FIR02470
C                                                                       FIR02480
C     DIMENSION  XS(4),XU(2),XL(2)                                      FIR02490
CC                                                                      FIR02500
CC  STARTING GUESS                                                      FIR02510
C     XS(1) = 1.0                                                       FIR02520
C     XS(2) = 0.5                                                       FIR02530
CC  UPPER LIMITS                                                        FIR02540
C     XU(1) = 6.0                                                       FIR02550
C     XU(2) = 6.0                                                       FIR02560
CC  LOWER LIMITS                                                        FIR02570
C     XL(1) = 0.0                                                       FIR02580
C     XL(2) = 0.0                                                       FIR02590
CC                                                                      FIR02600
C     R = 9./13.                                                        FIR02610
C     NTA = 5000                                                        FIR02620
C     NPR = 50                                                          FIR02630
C     NAV = 2                                                           FIR02640
C     NV = 2                                                            FIR02650
C     IP = 0                                                            FIR02660
CC                                                                      FIR02670
C      CALL BOXPLX (NV,NAV,NPR,NTA,R,XS,IP,XU,XL,YMN,IER)               FIR02680
C     WRITE(6,1) ((XS(I),I=1,4),YMN,IER)                                FIR02690
C   1 FORMAT (////,'   THE POINT IS LOCATED AT (XS(I)=) ',4(E13.7,5X),  FIR02700
C    1//,'   AND THE FUNCTION VALUE IS ',E13.7,' IER = ',I5)            FIR02710
CC                                                                      FIR02720
C     STOP                                                              FIR02730
C     END                                                               FIR02740
CC                                                                      FIR02750
```

FILE: FIR4P FORTRAN A1

```
CC                                                                      FIR02760
C     FUNCTION  KE(X)                                                   FIR02770
CC  EVALUATE CONSTRAINTS.  SET KE=0 IF NO IMPLICIT CONSTRAINT IS        FIR02780
CC  VIOLATED, OR SET KE=1 IF ANY IMPLICIT CONSTRAINT IS VIOLATED.       FIR02790
C     DIMENSION X(4)                                                    FIR02800
C     X1 = X(1)                                                         FIR02810
C     X2 = X(2)                                                         FIR02820
C     KE = 0                                                            FIR02830
C     X(3) = X1 + 1.732051*X2                                           FIR02840
C     IF (X(3) .LT. 0. .OR. X(3) .GT. 6.) GO TO 1                       FIR02850
C     X(4) = X1/1.732051 -X2                                            FIR02860
C     IF (X(4) .GE. 0.)  RETURN                                         FIR02870
CC                                                                      FIR02880
C   1 KE = 1                                                            FIR02890
C     RETURN                                                            FIR02900
C     END                                                               FIR02910
CC                                                                      FIR02920
CC.                                                                     FIR02930
C     FUNCTION  FE(X)                                                   FIR02940
C     DIMENSION X(4)                                                    FIR02950
CC                                                                      FIR02960
CC  THIS IS THE OBJECTIVE FUNCTION.                                     FIR02970
C     FE= -(X(2)**3 *(9.-(X(1)-3.)**2)/(46.76538))                      FIR02980
C     RETURN                                                            FIR02990
C     END                                                               FIR03000
C                                                                       FIR03010
C     METHOD                                                            FIR03020
C                                                                       FIR03030
C        THE COMPLEX METHOD IS AN EXTENSION AND ADAPTION OF THE SIM-    FIR03040
C        PLEX METHOD OF LINEAR PROGRAMMING.  STARTING WITH ANY ONE      FIR03050
C        FEASIBLE POINT IN N-DIMENSION SPACE A "COMPLEX" OF 2*N         FIR03060
C        VERTICES IS CONSTRUCTED BY SELECTING RANDOM POINTS WITHIN THE  FIR03070
C        FEASIBLE REGION.  FOR THIS PURPOSE N COORDINATES ARE FIRST     FIR03080
C        RANDOMLY CHOSEN WITHIN THE SPACE BOUNDED BY EXPLICIT CON-      FIR03090
C        STRAINTS.  THIS DEFINES A TRIAL INITIAL VERTEX.  IT IS THEN    FIR03100
C        CHECKED FOR POSSIBLE VIOLATION OF IMPLICIT CONSTRAINTS.  IF    FIR03110
C        ONE OR MORE ARE VIOLATED, THE TRIAL INITIAL VERTEX IS DISPLACEDFIR03120
C        HALF OF ITS DISTANCE FROM THE CENTROID OF PREVIOUSLY SELECTED  FIR03130
C        INITIAL VERTICES.  IF NECESSARY THIS DISPLACEMENT PROCESS IS   FIR03140
C        REPEATED UNTIL THE VERTEX HAS BECOME FEASIBLE.  IF THIS FAILS  FIR03150
C        TO HAPPEN AFTER 5*N+10 DISPLACEMENTS, THE SOLUTION IS ABAND-   FIR03160
C        ONED.  AFTER EACH VERTEX IS ADDED TO THE COMPLEX, THE CURRENT  FIR03170
C        CENTROID IS CHECKED FOR FEASIBILITY.  IF IT IS INFEASIBLE,     FIR03180
C        THE LAST TRIAL VERTEX IS ABANDONED AND AN EFFORT TO GENERATE   FIR03190
C        AN ALTERNATIVE TRIAL VERTEX IS MADE.  IF 5*N+10 VERTICES ARE   FIR03200
C        ABANDONED CONSECUTIVELY, THE SOLUTION IS TERMINATED.           FIR03210
C                                                                       FIR03220
C        IF AN INITIAL COMPLEX IS ESTABLISHED, THE BASIC COMPUTATION    FIR03230
C        LOOP IS INITIATED.  THESE INSTRUCTIONS FIND THE CURRENT WORST  FIR03240
C        VERTEX, THAT IS, THE VERTEX WITH THE LARGEST CORRESPONDING     FIR03250
C        VALUE FOR THE OBJECTIVE FUNCTION, AND REPLACE THAT VERTEX BY   FIR03260
C        ITS OVER-REFLECTION THROUGH THE CENTROID OF ALL OTHER VERTICES.FIR03270
C        (IF THE VERTEX TO BE REPLACED IS CONSIDERED AS A VECTOR IN     FIR03280
C        N-SPACE, ITS OVER-REFLECTION IS OPPOSITE IN DIRECTION, IN-     FIR03290
C        CREASED IN LENGTH BY THE FACTOR 1.3, AND COLLINEAR WITH        FIR03300
```

```
C        THE REPLACED VERTEX AND CENTROID OF ALL OTHER VERTICES.)          FIR03310
C                                                                          FIR03320
C        WHEN AN OVER-REFLECTION IS NOT FEASIBLE OR REMAINS WORST, IT      FIR03330
C        IS CONSIDERED NOT-PERMISSIBLE AND IS DISPLACED HALFWAY TOWARD     FIR03340
C        THE CENTROID.  AFTER FOUR SUCH ATTEMPTS ARE MADE UNSUCCESSFULLY   FIR03350
C        EVERY FIFTH ATTEMPT IS MADE BY REFLECTING THE OFFENDING VERTEX    FIR03360
C        THROUGH THE PRESENT BEST VERTEX, INSTEAD OF THROUGH THE CEN-      FIR03370
C        TROID.  IF 5*N+10 DISPLACEMENTS AND OVER-REFLECTIONS OCCUR        FIR03380
C        WITHOUT A SUCCESSFUL (PERMISSIBLE) RESULT, THE CURRENT BEST       FIR03390
C        VERTEX IS TAKEN AS AN INITIAL FEASIBLE POINT FOR A RESTART        FIR03400
C        RUN OF THE COMPLETE PROCESS.  RESTARTING IS ALSO UNDERTAKEN       FIR03410
C        WHEN 6*NV+10 CONSECUTIVE TRIALS HAVE BEEN MADE WITH NO SIGNIF-    FIR03420
C        ICANT CHANGE IN THE VALUE OF THE OBJECTIVE FUNCTION.  IN ALL      FIR03430
C        CASES, RESTARTING IS INHIBITED IF THE LAST RESTART DID NOT        FIR03440
C        PRODUCE A SIGNIFICANT IMPROVEMENT IN THE MINIMUM ATTAINED.        FIR03450
C                                                                          FIR03460
C        IT IS RECOMMENDED THAT THE USER READ THE REFERENCE FOR            FIR03470
C        FURTHER USEFUL INFORMATION.  IT SHOULD BE NOTED THAT THE          FIR03480
C        ALGORITHM DEFINED THERE HAS BEEN ALTERED TO FIND THE              FIR03490
C        CONSTRAINED MINIMUM, RATHER THAN THE MAXIMUM.                     FIR03500
C                                                                          FIR03510
C                                                                          FIR03520
C                                                                          FIR03530
C     REMARKS                                                              FIR03540
C                                                                          FIR03550
C        THE INTEGER PROGRAMMING OPTION WAS ADDED TO THIS PROGRAM          FIR03560
C        AS SUGGESTED IN REFERENCE (2).  A MIXED INTEGER/CONTINUOUS        FIR03570
C        VARIABLE VERSION OF BOXPLX WOULD BE EASY TO CREATE BY DE-         FIR03580
C        CLARING "IP" TO BE AN ARRAY OF NV CONTROL VARIABLES WHERE IP      FIR03590
C        (I)=1 WOULD INDICATE THAT THE I-TH VARIABLE IS TO BE CONFINED     FIR03600
C        TO INTEGER VALUES.  EACH STATEMENT OF THE FORM 'IF (IP .EQ.       FIR03610
C        1)' ETC. WOULD THEN NEED TO BE ALTERED TO 'IF (IP(I) .EQ. 1)'     FIR03620
C        ETC., WHERE THE SUBSCRIPT IS APPROPRIATELY CHOSEN.  NORMALLY,     FIR03630
C        XU AND XL VALUES ARE ALTERED TO BE AN EPSILON 'WITHIN' ACTUAL     FIR03640
C        VALUES DECLARED BY THE USER.  THIS ADJUSTMENT IS NOT MADE         FIR03650
C        WHEN IP=1.                                                        FIR03660
C                                                                          FIR03670
C        NOTE:  NO NON-LINEAR PROGRAMMING ALGORITHM CAN GUARANTEE THAT     FIR03680
C        THE ANSWER FOUND IS THE GLOBAL MINIMUM, RATHER THAN JUST A        FIR03690
C        LOCAL MINIMUM.  HOWEVER, ACCORDING TO REF.2, THE COMPLEX          FIR03700
C        METHOD HAS AN ADVANTAGE IN THAT IT TENDS TO FIND THE GLOBAL       FIR03710
C        MINIMUM MORE FREQUENTLY THAN MANY OTHER NON-LINEAR PROGRAM-       FIR03720
C        MING ALGORITHMS.                                                  FIR03730
C                                                                          FIR03740
C        IT SHOULD BE NOTED THAT THE AUXILIARY VARIABLE FEATURE CAN        FIR03750
C        ALSO BE USED TO DEAL WITH PROBLEMS CONTAINING EQUALITY CON-       FIR03760
C        STRAINTS.  ANY EQUALITY CONSTRAINT IMPLIES THAT A GIVEN VAR-      FIR03770
C        IABLE IS NOT TRULY INDEPENDENT.  THEREFORE, IN GENERAL, ONE       FIR03780
C        VARIABLE INVOLVED IN AN EQUALITY CONSTRAINT CAN BE RENUMBERED     FIR03790
C        FROM THE SET OF NV INDEPENDENT VARIABLES AND ADDED TO THE SET     FIR03800
C        OF NAV AUXILIARY VARIABLES.  THIS USUALLY INVOLVES RENUMBERING    FIR03810
C        THE INDEPENDENT VARIABLES OF THE GIVEN PROBLEM.                   FIR03820
C                                                                          FIR03830
C     SUBROUTINES AND FUNCTIONS REQUIRED                                   FIR03840
C                                                                          FIR03850
```

FILE: FIR4P FORTRAN A1

```
C       SUBROUTINE 'BOUT' AND FUNCTION 'FBV' ARE INTEGRAL PARTS OF      FIR03860
C       THE BOXPLX PACKAGE.                                             FIR03870
C                                                                       FIR03880
C       TWO FUNCTIONS MUST BE SUPPLIED BY THE USER.  THE FIRST, KE(X),  FIR03890
C       IS USED TO EVALUATE THE IMPLICIT CONSTRAINTS.  SET KE=0 AT THE  FIR03900
C       BEGINING OF THE FUNCTION, THEN EVALUATE THE IMPLICIT CON-       FIR03910
C       STRAINTS.  IN THE EXAMPLE ABOVE, THE FIRST CONSTRAINT, X(3),    FIR03920
C       MUST BE WITHIN THE RANGE (0. .LE. X(3) .LE. 6.).  THE SECOND    FIR03930
C       CONSTRAINT X(4), MUST BE .GE. 0. .  IF EITHER CONSTRAINT IS     FIR03940
C       NOT WITHIN THESE BOUNDS, CONTROL IS TRANSFERRED TO STATEMENT 1, FIR03950
C       AND KE IS SET TO "1" AND CONTROL IS RETURNED TO BOXPLX.         FIR03960
C                                                                       FIR03970
C       THE SECOND FUNCTION THE USER MUST PROVIDE EVALUATES THE OB-     FIR03980
C       JECTIVE FUNCTION.  IT IS CALLED FE(X) AS SHOWN IN THE EXAM-     FIR03990
C       PLE ABOVE, AND FE MUST BE SET TO THE VALUE OF THE OBJECTIVE     FIR04000
C       FUNCTION CORRESPONDING TO CURRENT VALUES OF THE NV INDEPENDENT  FIR04010
C       VARIABLES IN ARRAY 'X'.                                         FIR04020
C                                                                       FIR04030
C     REFERENCES                                                        FIR04040
C                                                                       FIR04050
C       BOX, M. J., "A NEW METHOD OF CONSTRAINED OPTIMIZATION AND A     FIR04060
C       COMPARISON WITH OTHER METHODS", COMPUTER JOURNAL, 8 APR. '65,   FIR04070
C       PP. 45-52.                                                      FIR04080
C                                                                       FIR04090
C       BEVERIDGE G., AND SCHECHTER R., "OPTIMIZATION: THEORY AND       FIR04100
C       PRACTICE", MCGRAW-HILL, 1970.                                   FIR04110
C                                                                       FIR04120
C     PROGRAMMER                                                        FIR04130
C                                                                       FIR04140
C       R.R. HILLEARY 1/1966.                                           FIR04150
C       REVISED FOR SYSTEM 360 4/1967                                   FIR04160
C       CORRECTED  1/1969                                               FIR04170
C       REVISED/EXTENDED BY L.NOLAN/R.HILLEARY  2/1975                  FIR04180
C       CORRECTED  8/1976                                               FIR04190
C                                                                       FIR04200
C                                                                       FIR04210
C     ..................................................................FIR04220
C                                                                       FIR04230
C                                                                       FIR04240
      SUBROUTINE BOXPLX (NV,NAV,NPR,NTZ,RZ,XS,IP,BU,BL,YMN,IER)         FIR04250
C                                                                       FIR04260
      DIMENSION V(50,50), FUN(50), SUM(25), CEN(25), XS(NV), BU(NV), BL(FIR04270
     1NV)                                                               FIR04280
C                                                                       FIR04290
      KV = 5                                                            FIR04300
      EP = 1.E-6                                                        FIR04310
      NTA = 2000                                                        FIR04320
      IF (NTZ.GT.0) NTA = NTZ                                           FIR04330
      R = RZ                                                            FIR04340
      IF (R.LE.0..OR.R.GE.1.) R=1./3.                                   FIR04350
      NVT = NV+NAV                                                      FIR04360
C                                                                       FIR04370
C           TOTAL VARS, EXPLICIT PLUS IMPLICIT                          FIR04380
      NT = 0                                                            FIR04390
C           CURRENT TRIAL NO.                                           FIR04400
```

FILE: FIR4P FORTRAN A1

```
      NPT = 0                                                          FIR04410
C           CURRENT NO. OF PERMISSIBLE TRIALS                          FIR04420
      NTFS = 0                                                         FIR04430
C           CURRENT NO. OF TIMES F HAS BEEN ALMOST UNCHANGED           FIR04440
C                                                                      FIR04450
C             CHECK FEASIBILITY OF START POINT                         FIR04460
C                                                                      FIR04470
      DO 4 I=1,NV                                                      FIR04480
      VT = XS(I)                                                       FIR04490
      IF (BL(I).LE.VT) GO TO 1                                         FIR04500
      II = -I                                                          FIR04510
      VT = BL(I)                                                       FIR04520
      GO TO 2                                                          FIR04530
    1 IF (BU(I).GE.VT) GO TO 3                                         FIR04540
      II = I                                                           FIR04550
      VT = BU(I)                                                       FIR04560
    2 IF (NPR.GT.0) WRITE (6,49) II                                    FIR04570
    3 V(I,1) = VT                                                      FIR04580
      CEN(I) = VT                                                      FIR04590
      IF (IP.EQ.1) GO TO 4                                             FIR04600
      BL(I) = BL(I)+AMAX1(EP,EP*ABS(BL(I)))                            FIR04610
      BU(I) = BU(I)-AMAX1(EP,EP*ABS(BU(I)))                            FIR04620
    4 SUM(I) = VT                                                      FIR04630
C                                                                      FIR04640
C                                                                      FIR04650
      NCE = 1                                                          FIR04660
C           NUMBER OF CONSTRAINT EVALUATIONS                           FIR04670
      I = 1                                                            FIR04680
      IF (KE(V(1,1)).EQ.0) GO TO 5                                     FIR04690
      IF (NPR.LE.0) GO TO 12                                           FIR04700
      WRITE (6,50)                                                     FIR04710
      GO TO 12                                                         FIR04720
    5 NFE = 1                                                          FIR04730
C                                                                      FIR04740
C   NUMBER OF VERTICES (K) = 2 TIMES NO. OF VARIABLES.                 FIR04750
      K = 2*NV                                                         FIR04760
C                                                                      FIR04770
C   NUMBER OF DISPLACEMENTS ALLOWED.                                   FIR04780
      NLIM = 5*NV+10                                                   FIR04790
C                                                                      FIR04800
C   NUMBER OF CONSECUTIVE TRIALS WITH UNCHANGED FE TO TERMINATE.       FIR04810
      NCT = NLIM+NV                                                    FIR04820
      ALPHA = 1.3                                                      FIR04830
      FK = K                                                           FIR04840
      FKM = FK-1.                                                      FIR04850
      BETA = ALPHA+1.                                                  FIR04860
C                                                                      FIR04870
C   INSURE SEED OF RANDOM NUMBER GENERATOR IS ODD.                     FIR04880
      IQR = R*1.E7                                                     FIR04890
      IF (MOD(IQR,2).EQ.0) IQR=IQR+101                                 FIR04900
C                                                                      FIR04910
C                     SET UP INITIAL VERTICES                          FIR04920
      FUN(1) = FE(V(1,1))                                              FIR04930
      YMN = FUN(1)                                                     FIR04940
    6 FI = 1.                                                          FIR04950
```

```
      FUNOLD = FUN(1)                                                   FIR04960
C                                                                       FIR04970
      DO 15 I=2,K                                                       FIR04980
      FI = FI+1.                                                        FIR04990
      LIMT = 0                                                          FIR05000
    7 LIMT = LIMT+1                                                     FIR05010
C                                                                       FIR05020
C   END CALCULATION IF FEASIBLE CENTROID CANNOT BE FOUND.               FIR05030
      IF (LIMT.GE.NLIM) GO TO 11                                        FIR05040
C                                                                       FIR05050
      DO 8 J=1,NV                                                       FIR05060
C                                                                       FIR05070
C   RANDOM NUMBER GENERATOR  (RANDU)                                    FIR05080
      IQR = IQR*65539                                                   FIR05090
      IF (IQR.LT.0) IQR = IQR+2147483647+1                              FIR05100
      RQX = IQR                                                         FIR05110
      RQX = RQX*.4656613E-9                                             FIR05120
      V(J,I) = BL(J)+RQX*(BU(J)-BL(J))                                  FIR05130
      IF (IP.EQ.1) V(J,I)=AINT(V(J,I)+.5)                               FIR05140
    8 CONTINUE                                                          FIR05150
C                                                                       FIR05160
      DO 10 L=1,NLIM                                                    FIR05170
      NCE = NCE+1                                                       FIR05180
      IF (KE(V(1,I)).EQ.0) GO TO 13                                     FIR05190
C                                                                       FIR05200
      DO 9 J=1,NV                                                       FIR05210
      VT = .5*(V(J,I)+CEN(J))                                           FIR05220
      IF (IP.EQ.1) VT = AINT(VT+.5)                                     FIR05230
      V(J,I) = VT                                                       FIR05240
    9 CONTINUE                                                          FIR05250
C                                                                       FIR05260
   10 CONTINUE                                                          FIR05270
C                                                                       FIR05280
   11 IF (NPR.LE.0) GO TO 12                                            FIR05290
      WRITE (6,51) I                                                    FIR05300
      CALL BOUT (NT,NPT,NFE,NCE,NV,NVT,V,I,FUN,CEN,I)                   FIR05310
   12 IER = -1                                                          FIR05320
      GO TO 48                                                          FIR05330
C                                                                       FIR05340
   13 DO 14 J=1,NV                                                      FIR05350
      SUM(J) = SUM(J)+V(J,I)                                            FIR05360
   14 CEN(J) = SUM(J)/FI                                                FIR05370
C                                                                       FIR05380
C   TRY TO ASSURE FEASIBLE CENTROID FOR STARTING.                       FIR05390
      NCE = NCE+1                                                       FIR05400
      IF (KE(CEN).EQ.0) GO TO 60                                        FIR05410
      SUM(J) = SUM(J) -V(J,I)                                           FIR05420
      GO TO 7                                                           FIR05430
   60 NFE = NFE+1                                                       FIR05440
      FUN(I) = FE(V(1,I))                                               FIR05450
   15 CONTINUE                                                          FIR05460
C                                                                       FIR05470
C   END OF LOOP SETTING OF INITIAL COMPLEX.                             FIR05480
      IF (NPR.LE.0) GO TO 17                                            FIR05490
      CALL BOUT (NT,NPT,NFE,NCE,NV,NVT,V,K,FUN,CEN,0)                   FIR05500
```

FILE: FIR4P FORTRAN A1

```
C                                                                       FIR05510
C   FIND THE WORST VERTEX, THE 'J'TH.                                   FIR05520
      J = 1                                                             FIR05530
C                                                                       FIR05540
      DO 16 I=2,K                                                       FIR05550
      IF (FUN(J).GE.FUN(I)) GO TO 16                                    FIR05560
      J = I                                                             FIR05570
   16 CONTINUE                                                          FIR05580
C                                                                       FIR05590
C   BASIC LOOP.  ELIMINATE EACH WORST VERTEX IN TURN.  IT MUST BECOME   FIR05600
C   NO LONGER WORST, NOT MERELY IMPROVED.  FIND NEXT-TO-WORST VERTEX,   FIR05610
C   THE 'JN'TH ONE.                                                     FIR05620
   17 JN = 1                                                            FIR05630
      IF (J.EQ.1) JN = 2                                                FIR05640
C                                                                       FIR05650
      DO 18 I=1,K                                                       FIR05660
      IF (I.EQ.J) GO TO 18                                              FIR05670
      IF (FUN(JN).GE.FUN(I)) GO TO 18                                   FIR05680
      JN = I                                                            FIR05690
   18 CONTINUE                                                          FIR05700
C                                                                       FIR05710
C   LIMT = NUMBER OF MOVES DURING THIS TRIAL TOWARD THE CENTROID        FIR05720
C    DUE TO FUNCTION VALUE.                                             FIR05730
      LIMT = 1                                                          FIR05740
C                                                                       FIR05750
C   COMPUTE CENTROID AND OVER REFLECT WORST VERTEX.                     FIR05760
C                                                                       FIR05770
      DO 19 I=1,NV                                                      FIR05780
      VT = V(I,J)                                                       FIR05790
      SUM(I) = SUM(I)-VT                                                FIR05800
      CEN(I) = SUM(I)/FKM                                               FIR05810
      VT = BETA*CEN(I)-ALPHA*VT                                         FIR05820
      IF (IP.EQ.1) VT = AINT(VT+.5)                                     FIR05830
C                                                                       FIR05840
C   INSURE THE EXPLICIT CONSTRAINTS ARE OBSERVED.                       FIR05850
   19 V(I,J) = AMAX1(AMIN1(VT,BU(I)),BL(I))                             FIR05860
C                                                                       FIR05870
      NT = NT+1                                                         FIR05880
C                                                                       FIR05890
C   CHECK FOR IMPLICIT CONSTRAINT VIOLATION.                            FIR05900
C                                                                       FIR05910
   20 DO 25 N=1,NLIM                                                    FIR05920
      NCE = NCE+1                                                       FIR05930
      IF (KE(V(1,J)).EQ.0) GO TO 26                                     FIR05940
C                                                                       FIR05950
C   EVERY 'KV'TH TIME, OVER-REFLECT THE OFFENDING VERTEX THROUGH THE    FIR05960
C   BEST VERTEX.                                                        FIR05970
      IF (MOD(N,KV).NE.0) GO TO 22                                      FIR05980
      CALL FBV (K,FUN,M)                                                FIR05990
C                                                                       FIR06000
      DO 21 I=1,NV                                                      FIR06010
      VT = BETA*V(I,M)-ALPHA*V(I,J)                                     FIR06020
      IF (IP.EQ.1) VT = AINT(VT+.5)                                     FIR06030
   21 V(I,J) = AMAX1(AMIN1(VT,BU(I)),BL(I))                             FIR06040
C                                                                       FIR06050
```

```
      GO TO 24                                                          FIR06060
C                                                                       FIR06070
C   CONSTRAINT VIOLATION:  MOVE NEW POINT TOWARD CENTROID.              FIR06080
C                                                                       FIR06090
   22 DO 23 I=1,NV                                                      FIR06100
      VT = .5*(CEN(I)+V(I,J))                                           FIR06110
      IF (IP.EQ.1) VT = AINT(VT+.5)                                     FIR06120
      V(I,J) = VT                                                       FIR06130
   23 CONTINUE                                                          FIR06140
C                                                                       FIR06150
   24 NT = NT+1                                                         FIR06160
   25 CONTINUE                                                          FIR06170
C                                                                       FIR06180
      IER = 1                                                           FIR06190
C                                                                       FIR06200
C   CANNOT GET FEASIBLE VERTEX BY MOVINC TOWARD CENTROID,               FIR06210
C   OR BY OVER-REFLECTING THRU THE BEST VERTEX.                         FIR06220
      IF (NPR.LE.0) CO TO 42                                            FIR06230
      WRITE (6,52) NT,J                                                 FIR06240
      CALL BOUT (NT,NPT,NFE,NCE,NV,NVT,V,K,FUN,CEN,J)                   FIR06250
      CO TO 42                                                          FIR06260
C                                                                       FIR06270
C   FEASIBLE VERTEX FOUND, EVALUATE THE OBJECTIVE FUNCTION.             FIR06280
   26 NFE = NFE+1                                                       FIR06290
      FUNTRY = FE(V(1,J))                                               FIR06300
C                                                                       FIR06310
C   TEST TO SEE IF FUNCTION VALUE HAS NOT CHANGED.                      FIR06320
      AFO = ABS(FUNTRY-FUNOLD)                                          FIR06330
      AMX = AMAX1(ABS(EP*FUNOLD),EP)                                    FIR06340
C                                                                       FIR06350
C   ACTIVATE THE FOLLOWINC TWO STATEMENTS FOR DIACNOSTIC PURPOSES ONLY. FIR06360
C     WRITE (6,99) J,AFO,AMX,FUNTRY,FUNOLD,FUN(J),FUN(JN),NTFS,N        FIR06370
C  99 FORMAT (1X,I3,6E15.7,2I5)                                         FIR06380
      IF (AFO.CT.AMX) CO TO 27                                          FIR06390
      NTFS = NTFS+1                                                     FIR06400
      IF (NTFS.LT.NCT) GO TO 28                                         FIR06410
      IER = 0                                                           FIR06420
      IF (NPR.LE.0) CO TO 42                                            FIR06430
      WRITE (6,53) K                                                    FIR06440
      CALL BOUT (NT,NPT,NFE,NCE,NV,NVT,V,K,FUN,CEN,0)                   FIR06450
      CO TO 42                                                          FIR06460
   27 NTFS = 0                                                          FIR06470
C                                                                       FIR06480
C   IS THE NEW VERTEX NO LONGER WORST?                                  FIR06490
   28 IF (FUNTRY.LT.FUN(JN)) GO TO 34                                   FIR06500
C                                                                       FIR06510
C   TRIAL VERTEX IS STILL WORST; ADJUST TOWARD CENTROID.                FIR06520
C   EVERY 'KV'TH TIME, OVER-REFLECT THE OFFENDINC VERTEX THROUGH THE    FIR06530
C   BEST VERTEX.                                                        FIR06540
      LIMT = LIMT+1                                                     FIR06550
      IF (MOD(LIMT,KV).NE.0) CO TO 30                                   FIR06560
      CALL FBV (K,FUN,M)                                                FIR06570
C                                                                       FIR06580
      DO 29 I=1,NV                                                      FIR06590
      VT = BETA*V(I,M)-ALPHA*V(I,J)                                     FIR06600
```

```
      IF (IP.EQ.1) VT = AINT(VT+.5)                                     FIR06610
   29 V(I,J) = AMAX1(AMIN1(VT,BU(I)),BL(I))                             FIR06620
C                                                                       FIR06630
      GO TO 32                                                          FIR06640
C                                                                       FIR06650
   30 DO 31 I=1,NV                                                      FIR06660
      VT = .5*(CEN(I)+V(I,J))                                           FIR06670
      IF (IP.EQ.1) VT = AINT(VT+.5)                                     FIR06680
      V(I,J) = VT                                                       FIR06690
   31 CONTINUE                                                          FIR06700
C                                                                       FIR06710
   32 IF (LIMT.LT.NLIM) GO TO 33                                        FIR06720
C                                                                       FIR06730
C   CANNOT MAKE THE 'J'TH VERTEX NO LONGER WORST BY DISPLACING TOWARD   FIR06740
C   THE CENTROID OR BY OVER-REFLECTING THRU THE BEST VERTEX.            FIR06750
      IER = 2                                                           FIR06760
      IF (NPR .LE. 0)  GO TO 42                                         FIR06770
      WRITE (6,52)  NT, J                                               FIR06780
      CALL BOUT (NT,NPT,NFE,NCE,NV,NVT,V,K,FUN,CEN,J)                   FIR06790
      GO TO 42                                                          FIR06800
   33 NT = NT+1                                                         FIR06810
      GO TO 20                                                          FIR06820
C                                                                       FIR06830
C   SUCCESS:  WE HAVE A REPLACEMENT FOR VERTEX J.                       FIR06840
   34 FUN(J) = FUNTRY                                                   FIR06850
      FUNOLD = FUNTRY                                                   FIR06860
      NPT = NPT+1                                                       FIR06870
C                                                                       FIR06880
C   EVERY 100'TH PERMISSIBLE TRIAL, RECOMPUTE CENTROID SUMMATION TO     FIR06890
C   AVOID CREEPING ERROR.                                               FIR06900
      IF (MOD(NPT,100).NE.0) GO TO 37                                   FIR06910
C                                                                       FIR06920
      DO 36 I=1,NV                                                      FIR06930
      SUM(I) = 0.                                                       FIR06940
C                                                                       FIR06950
      DO 35 N=1,K                                                       FIR06960
   35 SUM(I) = SUM(I)+V(I,N)                                            FIR06970
C                                                                       FIR06980
      CEN(I) = SUM(I)/FK                                                FIR06990
   36 CONTINUE                                                          FIR07000
C                                                                       FIR07010
      LC = 0                                                            FIR07020
      GO TO 39                                                          FIR07030
C                                                                       FIR07040
   37 DO 38 I=1,NV                                                      FIR07050
   38 SUM(I) = SUM(I)+V(I,J)                                            FIR07060
C                                                                       FIR07070
      LC = J                                                            FIR07080
C                                                                       FIR07090
   39 IF (NPR.LE.0) GO TO 40                                            FIR07100
      IF (MOD(NPT,NPR).NE.0) GO TO 40                                   FIR07110
C                                                                       FIR07120
      CALL BOUT (NT,NPT,NFE,NCE,NV,NVT,V,K,FUN,CEN,LC)                  FIR07130
C                                                                       FIR07140
C   HAS THE MAX. NUMBER OF TRIALS BEEN REACHED WITHOUT CONVERGENCE?     FIR07150
```

FILE: FIR4P FORTRAN A1

```
C   IF NOT, GO TO NEW TRIAL.                                                FIR07160
   40 IF (NT.GE.NTA) GO TO 41                                               FIR07170
C                                                                           FIR07180
C   NEXT-TO-WORST VERTEX NOW BECOMES WORST.                                 FIR07190
      J = JN                                                                FIR07200
      GO TO 17                                                              FIR07210
   41 IER = 3                                                               FIR07220
      IF (NPR.GT.0) WRITE (6,54)                                            FIR07230
C                                                                           FIR07240
C   COLLECTOR POINT FOR ALL ENDINGS.                                        FIR07250
C    1)  CANNOT DEVELOP FEASIBLE VERTEX.            IER = 1                 FIR07260
C    2)  CANNOT DEVELOP A NO-LONGER-WORST VERTEX.   IER = 2                 FIR07270
C    3)  FUNCTION VALUE UNCHANGED FOR K TRIALS.     IER = 0                 FIR07280
C    4)  LIMIT ON TRIALS REACHED.                   IER = 3                 FIR07290
C    5)  CANNOT FIND FEASIBLE VERTEX AT START.      IER = -1                FIR07300
   42 CONTINUE                                                              FIR07310
C                                                                           FIR07320
C  FIND BEST VERTEX.                                                        FIR07330
      CALL FBV (K,FUN,M)                                                    FIR07340
      IF (IER.GE.3) GO TO 44                                                FIR07350
C                                                                           FIR07360
C   RESTART IF THIS SOLUTION IS SIGNIFICANTLY BETTER THAN THE PREVIOUS,     FIR07370
C   OR IF THIS IS THE FIRST TRY.                                            FIR07380
      IF (NPR.LE.0) GO TO 43                                                FIR07390
      WRITE (6,55) (M,YMN,FUN(M))                                           FIR07400
   43 IF (FUN(M).GE.YMN) GO TO 47                                           FIR07410
      IF (ABS(FUN(M)-YMN).LE.AMAX1(EP,EP*YMN)) GO TO 47                     FIR07420
C                                                                           FIR07430
C   GIVE IT ANOTHER TRY UNLESS LIMIT ON TRIALS REACHED.                     FIR07440
   44 YMN = FUN(M)                                                          FIR07450
      FUN(1) = FUN(M)                                                       FIR07460
C                                                                           FIR07470
      DO 45 I=1,NV                                                          FIR07480
      CEN(I) = V(I,M)                                                       FIR07490
      SUM(I) = V(I,M)                                                       FIR07500
   45 V(I,1) = V(I,M)                                                       FIR07510
C                                                                           FIR07520
      DO 46 I=1,NVT                                                         FIR07530
   46 XS(I) = V(I,M)                                                        FIR07540
C                                                                           FIR07550
      IF (IER.LT.3) GO TO 6                                                 FIR07560
   47 IF (NPR.LE.0) GO TO 48                                                FIR07570
      CALL BOUT (NT,NPT,NFE,NCE,NV,NVT,V,K,FUN,V(1,M),-1)                   FIR07580
      WRITE (6,56) FUN(M)                                                   FIR07590
   48 RETURN                                                                FIR07600
C                                                                           FIR07610
   49 FORMAT (50H0INDEX AND DIRECTION OF OUTLYING VARIABLE AT START15)      FIR07620
   50 FORMAT (50H0IMPLICIT CONSTRAINT VIOLATED AT START. DEAD END. )        FIR07630
   51 FORMAT ('0CANNOT FIND FEASIBLE',I4,'TH VERTEX OR CENTROID AT STARTFIR07640
     1.')                                                                   FIR07650
   52 FORMAT (10H0AT TRIAL I4,54H CANNOT FIND FEASIBLE VERTEX WHICH IS NFIR07660
     10 LONGER WORST,I4,15X,'RESTART FROM BEST VERTEX.')                    FIR07670
   53 FORMAT (40H0FUNCTION HAS BEEN ALMOST UNCHANGED FOR I5,7H TRIALS)      FIR07680
   54 FORMAT (27H0LIMIT ON TRIALS EXCEEDED. )                               FIR07690
   55 FORMAT ('0BEST VERTEX IS NO.',I3,'  OLD MIN WAS ',E15.7,              FIR07700
```

FILE: FIR4P FORTRAN A1

```
     1 '  NEW MIN IS ',E15.7)                                                FIR07710
    56 FORMAT ('0MIN OBJECTIVE FUNCTION IS ',E15.7)                          FIR07720
       END                                                                   FIR07730
       SUBROUTINE FBV (K,FUN,M)                                              FIR07740
       DIMENSION FUN(50)                                                     FIR07750
       M = 1                                                                 FIR07760
C                                                                            FIR07770
       DO 1 I=2,K                                                            FIR07780
       IF (FUN(M).LE.FUN(I)) GO TO 1                                         FIR07790
       M = I                                                                 FIR07800
     1 CONTINUE                                                              FIR07810
C                                                                            FIR07820
       RETURN                                                                FIR07830
       END                                                                   FIR07840
       SUBROUTINE BOUT (NT,NPT,NFE,NCE,NV,NVT,V,K,FN,C,IK)                   FIR07850
       DIMENSION V(50,50), FN(50), C(25)                                     FIR07860
       WRITE (6,4) NT,NPT,NFE,NCE                                            FIR07870
C                                                                            FIR07880
       DO 1 I=1,K                                                            FIR07890
       WRITE (6,5) FN(I),(V(J,I),J=1,NV)                                     FIR07900
       IF (NVT.LE.NV) GO TO 1                                                FIR07910
       NVP = NV+1                                                            FIR07920
       WRITE (6,6) (V(J,I),J=NVP,NVT)                                        FIR07930
     1 CONTINUE                                                              FIR07940
C                                                                            FIR07950
       IF (IK.NE.0) GO TO 2                                                  FIR07960
C                                                                            FIR07970
       WRITE (6,7) (C(I),I=1,NV)                                             FIR07980
       RETURN                                                                FIR07990
     2 IF (IK.GE.0) GO TO 3                                                  FIR08000
       WRITE (6,8) (C(I),I=1,NV)                                             FIR08010
       RETURN                                                                FIR08020
     3 WRITE (6,9) IK,(C(I),I=1,NV)                                          FIR08030
       RETURN                                                                FIR08040
C                                                                            FIR08050
     4 FORMAT ('0NO. TOTAL TRIALS = ',I5,4X,'NO. FEASIBLE TRIALS = ',        FIR08060
      1I5,4X,'NO. FUNCTION EVALUATIONS = ',I5,4X,'NO. CONSTRAINT EVALUATI    FIR08070
      2ONS = ',I5/'0     FUNCTION VALUE',6X,'INDEPENDENT VARIABLES/DEPEND    FIR08080
      3ENT OR IMPLICIT CONSTRAINTS')                                         FIR08090
     5 FORMAT (1H ,E18.7,2X,7E14.7/(21X,7E14.7))                             FIR08100
     6 FORMAT (21X,7E14.7)                                                   FIR08110
     7 FORMAT (10HOCENTROID 11X,7E14.7/(21X,7E14.7))                         FIR08120
     8 FORMAT ('0  BEST VERTEX',7X,7E14.7/(21X,7E14.7))                      FIR08130
     9 FORMAT ('0CENTROID LESS VX',I2,2X,7E14.7/(21X,7E14.7))                FIR08140
       END                                                                   FIR08150
        FUNCTION FE(X)                                                       FIR08160
        DIMENSION X(4)                                                       FIR08170
        REAL J                                                               FIR08180
        COMMON J                                                             FIR08190
        CALL FIR(X)                                                          FIR08200
        FE=J                                                                 FIR08210
        RETURN                                                               FIR08220
        END                                                                  FIR08230
        FUNCTION KE(X)                                                       FIR08240
        DIMENSION X(4)                                                       FIR08250
```

```
      KE=0                                                          FIR08260
      RETURN                                                        FIR08270
      END                                                           FIR08280
```

FILE: FIR4SIMP FORTRAN A1

```
C*************************************************************************FIR00010
C                                                                         *FIR00020
C                              APPENDIX K                                 *FIR00030
C                                                                         *FIR00040
C                         FORTRAN PROGRAM FIR4SIM                         *FIR00050
C                  ADAPTIVE TRANSVERSAL FILTER SIMULATION                 *FIR00060
C                                                                         *FIR00070
C  TO PERFORM THE SIMULATION WE EMPLOY THE PROGRAM FIR WHICH              *FIR00080
C  WAS USED BY FIR4 TO CALCULATE THE FILTER OUTPUT VALUES WHEN            *FIR00090
C  CALCULATING THE OPTIMAL FILTER WEIGHTS.  WE ACCOMPLISH THIS            *FIR00100
C  BY CHANGING WS(*) TO THE ACTUAL W(*) AND ALSO DELETING THE             *FIR00110
C  INITIAL TRIAL BOUNDS REPRESENTED BY WU(*) AND WL(*).                   *FIR00120
C                                                                         *FIR00130
C*************************************************************************FIR00140
      REAL W(4),R,JE                                                      FIR00150
C  W(I) IS THE CALCULATED OPTIMAL GAIN                                    FIR00160
      W(1)=-7.5060358                                                     FIR00170
      W(2)=7.5403662                                                      FIR00180
      W(3)=4.7097464                                                      FIR00190
      W(4)=-5.3987589                                                     FIR00200
      WRITE (6,25)                                                        FIR00210
25    FORMAT(1X,' OPTIMAL GAINS',//)                                      FIR00220
      DO 30 I=1,4                                                         FIR00230
30       WRITE(6,40)I,W(I)                                                FIR00240
40    FORMAT(1X,'W(',I2,')=',F14.7)                                       FIR00250
      CALL FIR(W)                                                         FIR00260
      STOP                                                                FIR00270
      END                                                                 FIR00280
C*************************************************************************FIR00290
      SUBROUTINE FIR(XX)                                                  FIR00300
C  SUBROUTINE FIR(XX) SIMULATES THE FIR ADAPTIVE TRANSVERSAL FILTER       FIR00310
      COMMON J                                                            FIR00320
      REAL*8 J,W0,W1,W2,W3,X1,X2,X3,INPUT,OUTPUT                          FIR00330
      DIMENSION XX(4),DESIRE(105)                                         FIR00340
C  INITIAL CONDITIONS                                                     FIR00350
      ETIME=100.                                                          FIR00360
      T=0.0                                                               FIR00370
      ICOUNT=2                                                            FIR00380
C  INITIALIZE THE COST (CUMULATIVE ERROR) FUNCTION                        FIR00390
      J=0.0                                                               FIR00400
C  GAIN COEFFICIENTS TO BE OPTIMIZED                                      FIR00410
      W0=XX(1)                                                            FIR00420
      W1=XX(2)                                                            FIR00430
      W2=XX(3)                                                            FIR00440
      W3=XX(4)                                                            FIR00450
C  SHIFT REGISTERS                                                        FIR00460
      X1=0.0                                                              FIR00470
      X2=0.0                                                              FIR00480
      X3=0.0                                                              FIR00490
C  SIMULATE DESIRED OUTPUT SIGNAL                                         FIR00500
      DO 15 I=1,105                                                       FIR00510
15       DESIRE(I)=-1.0                                                   FIR00520
      DO 16 I=1,11                                                        FIR00530
16       DESIRE(I+44)=1.0                                                 FIR00540
C                                                                         FIR00550
```

FILE: FIR4SIMP FORTRAN A1

```
      K=1                                                                FIR00560
C  OUTPUT HEADING                                                       FIR00570
      WRITE(6,99)                                                        FIR00580
   99 FORMAT(' ','FIR TRANSVERSAL FILTER SIMULATION RESULTS',///,        FIR00590
     &'  TIME      INPUT      SIMULATED OUTPUT      DESIRED OUTPUT',/)   FIR00600
C  LOOP FOR 100 SAMPLE ITERATIONS                                        FIR00610
  200 CONTINUE                                                           FIR00620
C  SIMULATED INPUT SIGNAL (600 HZ + 1200 HZ + 1800 HZ)                   FIR00630
      INPUT=SIN(.1*T)*COS(.1*T)*(2.+COS(.1*T))                           FIR00640
C  SIMULATED OUTPUT SIGNAL FROM FIR FILTER                               FIR00650
      OUTPUT=W0*INPUT+W1*X1+W2*X2+W3*X3                                  FIR00660
C  WHEN TO PRINTOUT                                                      FIR00670
      IF (ICOUNT.EQ. 2) GO TO 50                                         FIR00680
      GO TO 300                                                          FIR00690
C  PRINTOUT                                                              FIR00700
   50 CONTINUE                                                           FIR00710
C  EASYPLOT OUTPUT OPTION                                                FIR00720
      WRITE (6,100) T,INPUT,OUTPUT,DESIRE(K)                             FIR00730
  100 FORMAT(2X,F8.4,1X,F8.4,3X,F8.4,13X,F8.4)                           FIR00740
C  SCREEN OUTPUT OPTION                                                  FIR00750
C     WRITE (6,100) T,INPUT,OUTPUT,DESIRE(K)                            FIR00760
C 100 FORMAT(1X,'TIME=',F7.3,5X,'INPUT=',F8.4,5X,'OUTPUT=',F8.4,5X,     FIR00770
C    &'DESIRED OUTPUT=',F8.4)                                           FIR00780
      ICOUNT=1                                                           FIR00790
C  TEST IF WANT TO STOP                                                  FIR00800
  300  IF (T.GE.ETIME) GO TO 400                                         FIR00810
C  JE=ERROR FUNCTION                                                     FIR00820
      JE=(OUTPUT-DESIRE(K))**2                                           FIR00830
C  J=COST FUNCTION (CUMULATIVE ERROR)                                    FIR00840
      J=J+JE                                                             FIR00850
C  STEP SIZE DELT                                                        FIR00860
      DELT=1.0                                                           FIR00870
      T=T+DELT                                                           FIR00880
      K=K+1                                                              FIR00890
      ICOUNT=ICOUNT+1                                                    FIR00900
      X3=X2                                                              FIR00910
      X2=X1                                                              FIR00920
      X1=INPUT                                                           FIR00930
      GO TO 200                                                          FIR00940
  400 RETURN                                                             FIR00950
      END                                                                FIR00960
```

LIST OF REFERENCES

1. Get Away Special (GAS) Small Self-Contained Payloads Experimenter Handbook, pp. 7-9, National Aeronautics and Space Administration, Goddard Space Flight Center, 1984.

2. AIAA/SAE Joint Propulsion Conference Report 78-1007, Space Shuttle Orbiter Auxiliary Power Unit Configuration and Performance, by J. R. Baughman and M. W. Reck, pp. 2-3, 25 July 1978.

3. NASA DATE Report 003, Payload Bay Acoustic and Vibration Data from STS-2 Flight, pp. 1-1 through 1-4 and 1-11 through 1-23, January 1982.

4. NASA DATE Report 005, Payload Bay Acoustic and Vibration Data from STS-4 Flight, pp. 1-1 through 1-4 and 1-11 through 1-23, December 1982.

5. Chen, C-T., One Dimensional Digital Signal Processing, pp. 141-326, Marcel Dekker, 1979.

6. Johnson, D. E., Johnson, J. R., and Moore, H. P., A Handbook of Active Filters, pp. 11-68, Prentice-Hall, 1980.

7. Intel Corp., The 2920 Analog Signal Processor Design Handbook, Intel, Santa Clara, 1979.

8. Gold, B., and Rabiner, L. R., Theory and Application of Digital Signal Processing, pp. 219-224, Prentice-Hall, 1975.

9. Intel Corp. AR-81, Single Chip N-MOS Microcomputer Processes Signals in Real Time, by M. E. Hoff and M. Townsend, pp. 1-8, March 1979.

10. Intel Corp., SDK-2920 System Design Kit User's Guide, Intel, Santa Clara, 1983.

11. Intel Corp., 2920 Assembly Language Manual, Intel, Santa Clara, 1979.

12. Naval Undersea Center TP 530, Principles and Applications of Adaptive Filters: A Tutorial Review, by J. M. McCool, pp. 1-11, March 1977.

13. Parker, S. R., An Integrated Approach to Discrete Signal Processing for Scientists and Engineers (Pre-publication notes), Naval Postgraduate School, Monterey, California, 1983.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Technical Information Center Cameron Station Alexandria, Virginia 22304-6145	2
2.	Library, Code 0142 Naval Postgraduate School Monterey, California 93943-5100	2
3..	Commander, Naval Space Command Dahlgren, Virginia 22448	1
4.	Professor Rudolf Panholzer, Code 62Pz Department of Electrical and Computer Engineering Naval Postgraduate School Monterey, California 93943-5100	3
5.	Professor Sydney R. Parker, Code 62Px Department of Electrical and Computer Engineering Naval Postgraduate School Monterey, California 93943-5100	1
6.	Dr. Bharat Madan, Code 62Px Department of Electrical and Computer Engineering Naval Postgraduate School Monterey, California 93943-5100	1
7.	Department Chairman, Code 62 Department of Electrical and Computer Engineering Naval Postgraduate School Monterey, California 93943	1
8.	LCDR Dorsett W. Jordan Naval Research Laboratory (Code 9110-52) 4555 Overlook Ave. S. W. Washington, D. C. 20375	2
9.	Mr. Donald E. Jordan 630 Steamboat Road Apartment 4E-N Greenwich, Connecticut 06830	1

10. Mr. Dorsett M. Jordan 1
2041 Shillingwood Drive
Kennesaw, Georgia 30144

11. Mrs. Victoria Soler 1
16 Weld Street
West Roxbury, Massachusetts 02131

214656

Thesis
J754 Jordan
c.1 A matched filter
algorithm for acoustic
signal detection.

22 FEB 90 35094
21 FEB 91 80541

214656

Thesis
J754 Jordan
c.1 A matched filter
algorithm for acoustic
signal detection.

AMERICAN BINDERIES

thesJ754
A matched filter algorithm for acoustic
3 2768 000 67670 4
DUDLEY KNOX LIBRARY